Automated Parameter Tuning in Cell Tracking: Evaluating Bayesian and Evolutionary Approaches

Tim August B. Huygelen[1]

[1]From the Department of Structural and Molecular Biology, Division of Biosciences, University College London (UCL), London, UK WC1E 6BT

To whom correspondence should be addressed: Tim August B. Huygelen, +44 07547311095, tim.huygelen.20@ucl.ac.uk.

Keywords: cell biology, computational biology, bioinformatics, microscopic imaging, cell migration

ABSTRACT

Accurate cell tracking is critical for understanding cellular dynamics in bioimaging. However, manual parameter tuning of cell tracking software, such as btrack, is labour intensive and inefficient. This study investigates the automation of parameter tuning using advanced algorithms, including Bayesian Optimisation (BO), Tree-structured Parzen Estimator (TPE), Covariance Matrix Adaptation Evolution Strategy (CMA-ES), Non-dominated Sorting Genetic Algorithm II (NSGA-II), and random search implemented within the Optuna framework. We utilised the traccuracy cell tracking evaluation software to assess the performance improvements achieved through automated tuning on 15 Cell Tracking Challenge datasets.

We demonstrated that automated parameter tuning significantly reduces the reliance on manual input, improving the reliability and accuracy of cell tracking under various conditions. Comparative analysis showed that TPE frequently achieved the best overall tracking score, while NSGA-II excelled in double-objective optimisation, with random search performing competitively.

Our research highlights the potential of parameter optimisation techniques to improve the reliability and accuracy of cell tracking software. We emphasise the need for a large, standardised cell microscopy database to train more robust cell tracking tools. By automating the parameter tuning process, we aim to bridge the gap between technological potential and practical application of cell tracking software, ultimately contributing to a deeper understanding of cellular dynamics and improving the monitoring of biological processes in physiologically relevant environments. This study sets the stage for further research into the optimisation of automated hyperparameter tools, such as BO, in the field of cell tracking, and advocates their wider adoption by the cell and molecular biology communities.

Cell Tracking

Cell tracking, to track individual cells over time and space, is essential for understanding the dynamic processes that govern cell behaviour, such as cell division, migration and differentiation. Computational tools for automated cell tracking have significantly advanced our understanding of cell biology by enabling the analysis of large datasets of time-lapse microscopy images. However, current tracking software often requires tedious manual parameter tuning to achieve accurate results. In this study, we present a novel software implementation to automate the parameter tuning process for the btrack cell tracking software (1, 2) using Bayesian optimisation (BO), evolutionary strategies and random search. We aim to evaluate these methods to find a suitable strategy for automated parameter tuning of cell tracking software.

History of Cell Tracking

The journey of cell tracking began with Robert Hooke's invention of the microscope in 1665, marking the first description of 'cells' as compartment-like structures in cork (3). Building on Hooke's discovery, Antonie van Leeuwenhoek made the first observations and descriptions of bacteria, protozoa and cell movement (4, 5).

The emergence of computational cell tracking can be traced back to the pioneering work of Davenport et al. who used a rudimentary digital computer and video setup, termed the "bugwatcher", to analyse the behaviour of microorganisms, marking the first example of a primitive digital computer for automated single-cell tracking (6). Despite its reliance on considerable manual input, this early attempt was a significant first step for computational tools in the field.

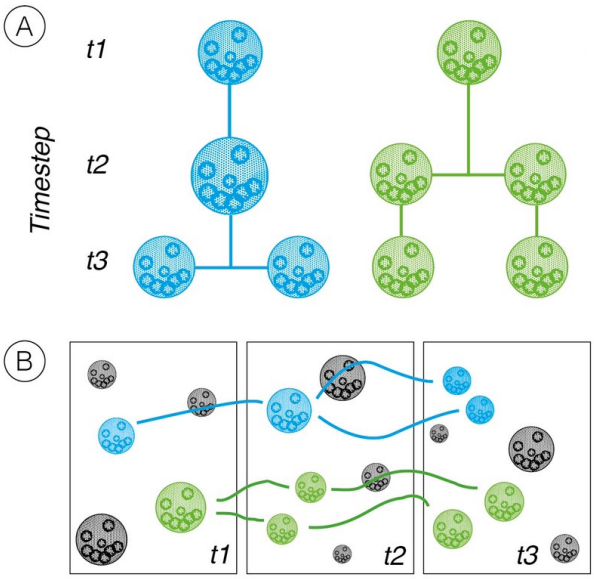

Figure 1: Temporal and Spatial Cell Tracking. (A) Visualisation of two simple cell lineages, with blue and green colours indicating different lineages, each branching at unique time points. (B) Schematic of hypothetical microscopy images taken at three time points. The behaviour of the highlighted blue and green cells at each time point corresponds to the lineage trees shown in (A).

In the late 2000s, robust open-source computational cell tracking algorithms such as CellProfiler (7) emerged, providing researchers with accessible tools for cell tracking. By integrating advances in computer vision and image processing, these tools allowed cells to be tracked across sequences of digital images (8, 9).

Despite these advances, including the integration of machine learning, there remains a gap between technological potential and practical applications. Deep learning has improved accuracy and scalability, but the heavy reliance on human input remains, and computational tools have yet to consistently outperform human experts (10–14). Traditional tools still perform comparably in the Cell Tracking Challenge, highlighting the need for further development (15–18). The limited success of deep learning is due to the lack of large, standardised datasets (19, 20).

Many cell tracking tools require careful tuning of user-defined parameters to make accurate predictions (2, 21, 22). This manual tuning is time consuming and difficult as optimal values are not immediately obvious (23). Reducing this dependency could improve the monitoring of cell behaviour in physiologically relevant environments (24, 25), thereby improving our understanding of cell behaviour and cancer progression (26, 27).

Automated parameter optimisation, using frameworks such as Optuna (28) and Ray Tune (29), has been used to tune the parameters of cell tracking software (30), resulting in improved tracking accuracy. Parameter optimisation algorithms, including BO and genetic algorithms, have shown significant improvements in cell segmentation software (31). Automated tuning has also been successful in non-biological tracking software (32).

btrack

btrack is a cell tracking software that combines deep learning models with Bayesian tracking algorithms to construct lineage trees from time-lapse cell microscopy (1, 2). The software consists of two main components, a motion model and a hypothesis model, both of which depend on a set of parameters that determine how they make their cell linking and track connection decisions, respectively.

The motion model constructs tracklets by linking cell observations over time without considering cell division. It uses a Bayesian belief matrix with a uniform prior for track association probabilities, updated with information from Kalman filters. This matrix incorporates predictions and cell state information to update the probability of track continuation or loss. Track hypotheses are generated and evaluated using motion and appearance data to optimise global track configurations, allowing new tracks to be initialised or lost tracks to be identified based on the calculated posterior probabilities.

The hypothesis model assembles the tracks constructed by the motion model into fully formed lineage trees. Using multiple hypothesis testing, the algorithm proposes different fates for each track based on appearance and motion characteristics. Hypotheses include false positives, initialisation and termination at specific film locations, tracklet merging, splitting and apoptosis. The log-likelihoods of the hypotheses are computed and represented in a binary matrix that aligns hypotheses with track actions, allowing the optimisation of track configurations to maximise the likelihood function. The optimal set of hypotheses is determined, leading to the construction of the predicted lineage tree.

Optimisation Strategies

Optimising a set of parameters in an optimisation loop requires several components: a parameter search space, a function or problem to evaluate, metrics to quantify the performance of the function, and a sampler to suggest new parameters for evaluation. We will discuss several options for sampler algorithms in the following section, the other components we chose for our optimisation loop are described in detail in the EXPERIMENTAL PROCEDURES section.

Bayesian Optimisation (BO)

Brief History BO is a sequential decision strategy using Bayesian inference to guide the search for a function's optimum, focusing computational resources on the most promising areas. Originating in the 18th century with Thomas Bayes' theorem, which updates hypothesis probability as evidence is gathered, BO techniques saw significant development in the 1970s with Jonas Mockus's work. Mockus introduced BO for global optimisation problems, demonstrating its effectiveness with limited evaluations. Since the 2010s, BO has rapidly grown in applications and methodological advances. The successful adoption of Gaussian Processes (GPs) for objective function modelling was followed by the Tree-structured Parzen Estimator (TPE), popular for hyper-parameter optimisation. TPE outperforms GPs in computational efficiency and scalability, handling non-continuous and conditional search spaces (33, 34).

Tree-structured Parzen Estimator (TPE) TPEs have gained significant attention due to their efficient hyperparameter tuning capabilities compared to GPs (35, 36). TPEs uniquely handle the exploration-exploitation trade-off, pivotal in hyperparameter tuning. Unlike traditional BO methods relying on GPs to model the objective function, TPE employs a non-parametric, density estimation technique. This method constructs probabilistic models for "good" and "bad" hyperparameter sets using Kernel Density Estimation (KDE), allowing efficient search process direction. This contrasts with the more computationally intensive updating and inverting of covariance matrices in GPs, especially in high-dimensional spaces. TPE is particularly useful for long-term prediction in time-series data (37), classification in medical imaging (38), energy consumption forecasting (39), and genomic prediction (40). TPE effectively handles discrete, categorical, and conditional variables with lower computational complexity than GPs (34).

Evolutionary Algorithms

Having explored BO and TPE for hyperparameter optimisation, we now turn our attention to evolutionary strategies. These strategies provide a robust alternative by using principles inspired by natural evolution to search for optimal solutions to complex, single- or multi-objective optimization problems.

Non-dominated Sorting Genetic Algorithm II (NSGA-II) NSGA-II is a well-known and widely used evolutionary algorithm for multi-objective optimisation (41). NSGA-II uses a non-dominated sorting approach to guide the convergence of the population towards the Pareto front, and employs crowding distance to manage population diversity (42). The effectiveness of the algorithm has been demonstrated in solving multi-objective optimisation problems where other dominance-based selection algorithms may struggle to discriminate candidate solutions due to a large number of non-dominated solutions (43).

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) CMA-ES is a stochastic and derivative-free singleobjective optimiser that works by iteratively generating new candidate solutions around an adaptive recombination point (44). The algorithm is invariant with respect to the ranking of the estimated candidate solutions, which contributes to its robustness and effectiveness (45). It is particularly effective in solving difficult non-smooth, non-convex problems (46). The algorithm is known for its ability to learn dependencies between parameters, making it successful in optimising real parameter problems (47). CMA-ES has become a standard in continuous black-box evolutionary optimization and is considered one of the most advanced optimizers for such problems (48).

Aims

In this study we implement and investigate an automated parameter tuning framework, using Optuna (28) for hyperparameter optimisation of the btrack cell tracking software (1, 2). Comparative analysis will include BO via TPE and genetic algorithms, specifically CMA-ES (49) and NSGA-II (41). The effectiveness of these tuned parameters will be evaluated against the default settings of the btrack software, focusing on accuracy improvements.

RESULTS

This section details experiments examining the impact of various optimisation strategies and parameter configurations on the performance of the btrack cell tracking algorithm. The primary aims were to understand how pruning, the number of optimisation objectives, and different samplers influence the tracking efficiency and accuracy. Additionally, a two-stage optimisation process was evaluated, and cluster analysis was performed. The findings offer valuable insights into optimising cell tracking algorithms for enhanced performance across datasets.

Pruning

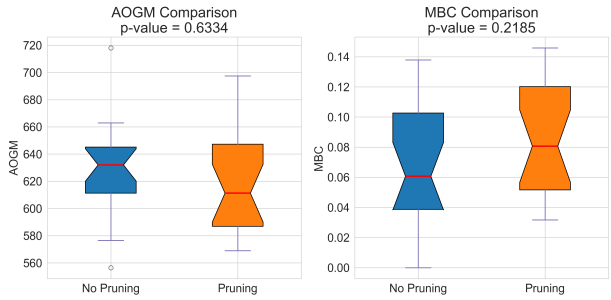

Figure 2: Box plot visualising the distribution of AOGM and MBC values with and without pruning. High AOGM and low MBC values indicate poorer performance.p-values were calculated using a two-sided t-test.

Rationale Pruning enhances computational efficiency by discarding poorly performing trials. This is particularly helpful when dealing with large datasets or complex optimisation problems. However, the impact of pruning on the overall performance of the optimisation process is not perfectly understood (50). When the btrack hyperparameters are badly configured, it takes longer for btrack to compute the tracking, so often long runtime is associated with a bad tracking outcome. We hypothesised that pruning based on trial runtime would be effective. We implemented a time-based pruning strategy to prematurely end trials that take longer than a specified time threshold. The aim was to assess whether pruning could improve the computational efficiency of the optimisation process without significantly affecting the quality of the results.

Outcomes Pruning with a 120-second timeout was assessed in 9 studies each with and without pruning (35 trials each using the TPE sampler) on the `BF-C2DL-MuSC` dataset. No significant difference was found in AOGM (p = 0.6334) or MBC (p = 0.2185) values. AOGM without pruning: mean 625.8750, SD 36.4666; with pruning: mean 620.3500, SD 36.2129. MBC without pruning: mean 0.0690, SD 0.0420; with pruning: mean 0.0846, SD 0.0366. Pruning reduced computation time by 14%, enhancing efficiency without affecting performance

Samplers and Number of Objectives

Rationale We compared samplers (TPE, CMA-ES, NSGA-II, and random search) to optimise the btrack algorithm, hypothesising TPE, CMA-ES, and NSGA-II would outperform random search, and TPE would outperform CMA-ES and NSGA-II. We also expected dual objectives to be more effective for the MBC metric. Each of the five parameter-objective combinations was tested in three studies across the 10 smallest datasets (50 studies total, 64 trials each). TPE and NSGA-II were used for dual objectives (AOGM and MBC), random search for these objectives, and CMA-ES and TPE for single objective (AOGM)

Analysis of Sampler Performance Figure 3 illustrates the performance across samplers. For AOGM, TPE (dual and single objectives) and NSGA-II (dual objective) had the lowest mean AOGM in two datasets each, while random search excelled in one dataset. CMA-ES did not achieve the best AOGM in any of the datasets. Statistical tests showed that NSGA-II significantly outperformed CMA-ES (t-test $p = 1.83 \times 10^{-5}$, Wilcoxon test $p = 3.84 \times 10^{-11}$). No significant differences were found between NSGA-II and random search overall (t-test $p = 0.965$, Wilcoxon test $p = 0.813$), while CMA-ES significantly underperformed compared to random search (t-test $p = 1.29 \times 10^{-5}$, Wilcoxon test $p = 2.51 \times 10^{-12}$).

For MBC, NSGA-II had the highest mean MBC in two datasets, with random search and TPE (dual objective) each excelling in one. Neither CMA-ES nor TPE (single objective) performed best in any dataset. Overall, NSGA-II significantly outperformed CMA-ES (t-test $p = 8.29 \times 10^{-7}$, Wilcoxon test $p = 1.59 \times 10^{-9}$). Comparisons between NSGA-II and random search showed no significant differences (t-test $p = 0.872$, Wilcoxon test $p = 0.257$), whereas random search significantly outperformed TPE (single objective) (t-test $p = 1.20 \times 10^{-5}$, Wilcoxon test $p = 1.92 \times 10^{-6}$).

Overall, TPE and NSGA-II outperformed CMA-ES and random search in several cases. TPE frequently achieved the lowest AOGM, while NSGA-II excelled in MBC. Contrary to our hypothesis, random search showed competitive performance while CMA-ES consistently underperformed.

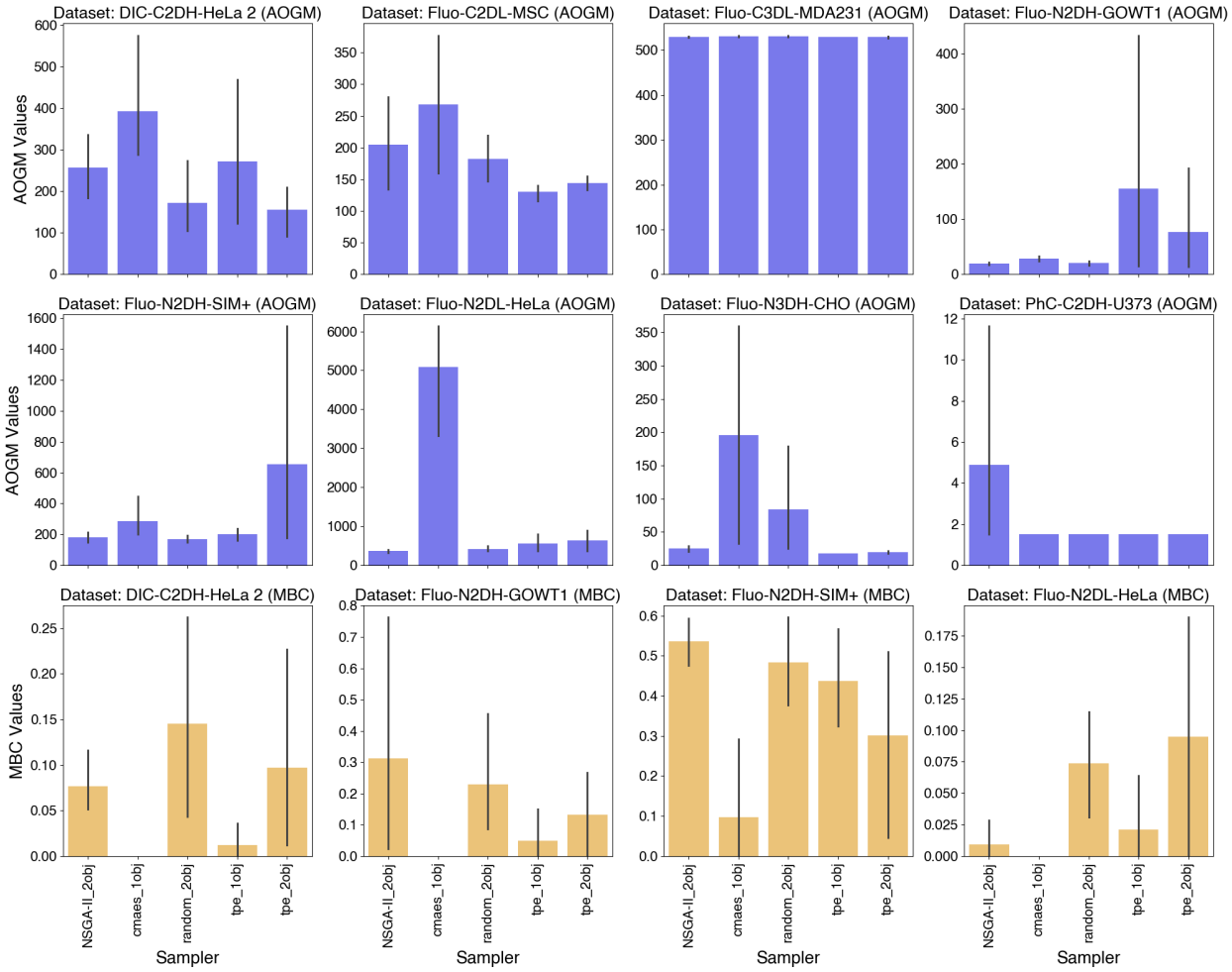

Figure 3: Box plots illustrating the distribution of AOGM and MBC values across different samplers. The samplers are referred to using the notation `tpe_2obj` for TPE with dual objective, `tpe_1obj` for TPE with a single objective, `cmaes_1obj` for CMA-ES with a single objective, `NSGA-II_2obj` for NSGA-II with dual objective, and `random_2obj` for random search with dual objective. Datasets where the samplers all have the same performance are not shown. Yellow box plots correspond to MBC, and blue box plots to AOGM.

Failure Rate Analysis We were surprised that no sampler consistently outperformed random search and noted the datasets where TPE performed poorly, namely (`C2DH-HeLa 2`, `Fluo-N2DH-GOWT1`, `Fluo-N3DH-SIM+`). We found that TPE's trials often reached timeout limits for pruning. This prompted us to examine the failure ratios due to timeouts among the samplers in these datasets. Figure 4 shows distribution of failure rates among samplers: CMA-ES (0.113), NSGA-II (0.219), Random (0.245), and TPE (0.489). A t-test between Random and TPE revealed a significant difference (t-statistic = -4.018, $p = 0.00054$). This suggests that TPE's poorer performance compared to Random search in these three datasets may be related to our pruning implementation which does not give negative feedback and thus allows the TPE to continue sampling unfavourable configurations ad infinitum.

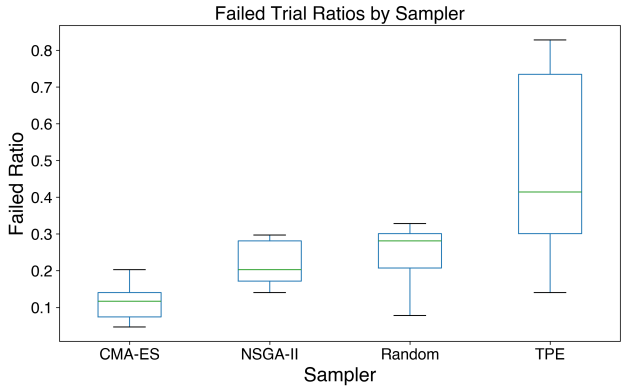

Figure 4: Comparison of the failure rates of different samplers. The failure rate is defined as the ratio of trials that did not complete within the time limit over the total number of trials.

Grid Search To assess the feasibility of grid search (51), another popular parameter search method, for our parameter optimisation problem, we calculated the number of trials required for a grid search approach with our 18 parameters. The results showed that even a minimalist setup of only three values per parameter (two for our Boolean parameter) would require over 258 million trials, highlighting the impracticality of this method for high-dimensional parameter spaces. A maximum feasible study of 243 trials would require reducing the parameter space to 5, while still being able to evaluate only 3 values per parameter.

Two-Step Approach

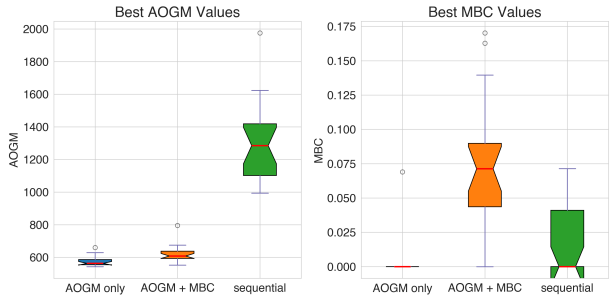

Figure 5: Comparison of AOGM and MBC values between the two-step and single-step optimisation approaches. High AOGM and low MBC values indicate poorer performance.

We hypothesised that sequential optimisation of btrack parameters would yield marginal improvements over a one-step approach. This approach, recommended in btrack's manual tuning guidelines, was explored for the possible effect of the btrack hypothesis model to mask an inadequately tuned motion model, resulting in an unsatisfactory tracking strategy. We conducted 20 studies per setup, each containing 32 trials, using the TPE sampler for the `BF-C2DL-MuSC` dataset. The two-step approach optimised motion model parameters first, then hypothesis model parameters, while the one-step approach optimised all parameters simultaneously.

Surprisingly, as shown in Figure 5, the two-step optimisation performed significantly worse than the one-step approach. The mean AOGM for the sequential optimisation was 1316.4750 (Std Dev = 252.6883), significantly higher than the single-step mean of 574.5750 (Std Dev = 30.3134), with a p-value of 0.000. The mean MBC for the sequential optimisation (0.0202, Std Dev = 0.0276) was lower than the AOGM + MBC approach (0.0733, Std Dev = 0.0473), with a p-value of 0.000.

Optimised Parameter Ranges per Dataset

Parameter distributions were analysed for three 128-trial studies with a double objective (AOGM and MBC) and one study with a single objective (AOGM) on all 15 selected datasets, using the TPE sampler. All pareto optimal trials and their corresponding parameters were used, Figure 6 indicated significant variation of optimal parameters distributions between dataset, once again underscoring that parameter tuning is essential for btrack to function as expected. The means and medians of each parameter were calculated across the best performing trials for each dataset. These values closely matched the default parameters in several instances, such as for `g_sigma` and `theta_dist`, while for other parameters such as `r_sigma` and `lambda_link` the optimal values differed significantly from the defaults for every dataset.

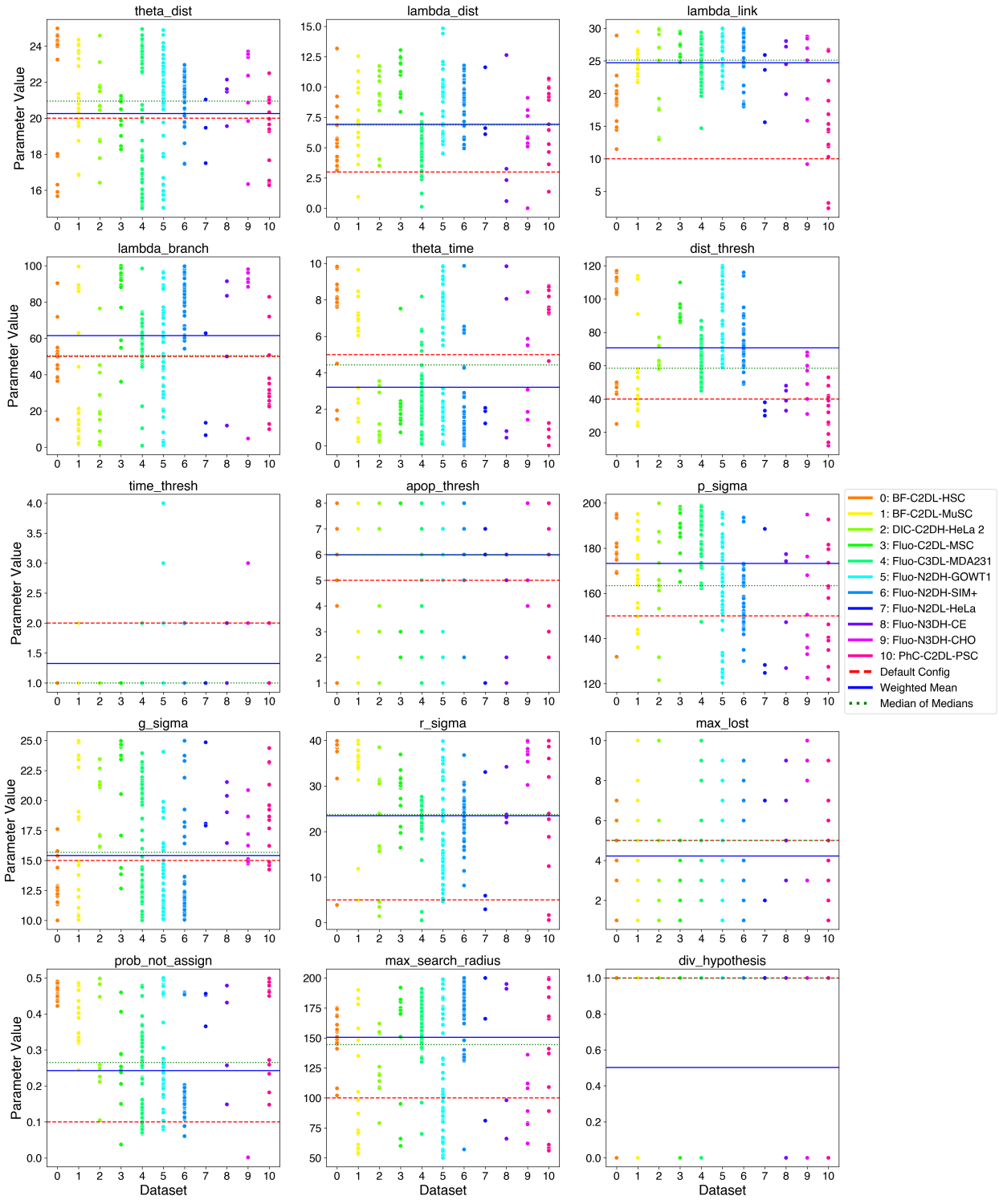

Figure 6: Scatter plots show the parameter distributions for the best performing trials across the datasets as identified by the TPE sampler. Each subplot corresponds to a specific parameter. Red dashed lines show default values, blue lines show the mean and green dashed lines show the median of the medians.

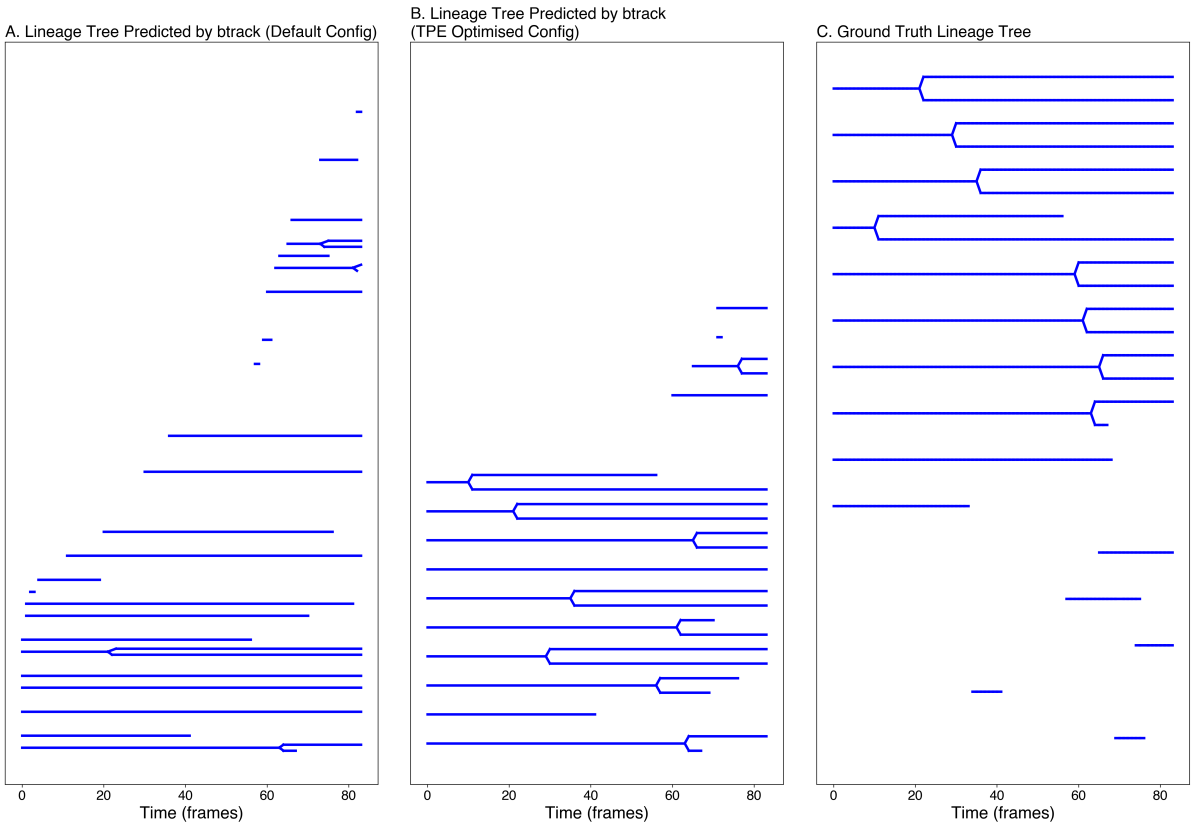

Figure 7: Comparison of lineage tree prediction before and after optimisation for the `DIC-C2DH-HeLa` dataset. (A) Lineage tree generated by btrack with default parameters (prior to optimisation). (B) Lineage tree obtained after 400 trials of dual-objective optimisation using the TPE sampler. (C) Reference ground truth lineage tree.

DISCUSSION

Optimisation Loop Setup for Cell Tracking

Some key considerations for those intending to use our software or intending to construct their own parameter optimisation framework for cell tracking software are as follows:

Sampler and Objectives Both efficiency and accuracy should be considered when selecting samplers and defining objectives for parameter optimisation in cell tracking software such as btrack. Different samplers such as TPE, CMA-ES and NSGA-II have different strengths. For example, TPE excels in modelling the parameter space and learning from previous trials, while NSGA-II is effective in dual objective optimisations involving metrics such as AOGM and MBC. Our study showed that although random search is often a baseline method, it performs competitively, highlighting its potential as an unbiased search method. In addition, a dual objective setup can balance tracking accuracy with mitotic event detection, although it increases complexity and the potential for broken tracklets. Single objective (AOGM) optimisations optimise global tracking accuracy but often miss mitotic branching. Therefore, the choice of sampler and number of objectives should be based on the specific objectives of the cell tracking task and the available computational resources.

pruning Pruning improves the efficiency of the optimisation process. However, the current implementation is somewhat memory inefficient, which limits its effectiveness on large datasets (cell counts per frame > 100). The pruning setup is flexible and can be adapted to user needs. Pruned trials can be marked as 'failed', causing the sampler to ignore the result in future iterations, or they can output a user-defined value to discourage similar parameter configurations.

Single Step vs. Two Step Optimisation While two-step optimisation makes intuitive sense, and the btrack documentation recommends that human users tune them separately, it appears to perform significantly worse than single-step optimisation in every metric tested. This is probably due to the fact that the two-step optimisation process is more complex and may not adapt well when pa-

rameters are interdependent. The single-step optimisation process is simpler and more efficient, and we recommend using it as the default setting for the optimisation loop.

Lack of validation on separate datasets

In our study, no sectioning of the data into training and validation datasets was performed. This could lead to overfitting and the optimal configuration may not generalise for its specific use case. Although this was beyond the scope of this study, future work should include validation on a separate dataset to ensure the robustness of the optimised parameters. We also recommend the same for those implementing this framework in their own research.

Novelty of Using BO for Cell Tracking

To date, no independent studies have been published that use BO specifically to tune the parameters of cell tracking software. Existing research in this area includes one study that uses max-margin structured learning and the bundle method for parameter optimisation (52), and another that uses BO to tune a general multi-object tracking algorithm (53). In addition, recent developments in deep reinforcement learning for cell tracking, while still limited in scope, provide a promising foundation for future exploration in this area (54).

Impact of parameter optimisation techniques on the wider biological community

BO and evolutionary algorithms for parameter optimisation can be incredibly powerful despite their relatively simple architecture. They provide a simple yet effective way to tune existing software based on accuracy measures of validated underlying truths, rather than relying on subjective human judgment. The wider cell and molecular biology community could benefit from exploring these methods to improve their existing software, particularly for tools where the current paradigm relies too heavily on individual users to tune a set of parameters. Software such as Optuna (28) and BoTorch (55), which were designed for machine learning optimisation, are efficient and well suited to a variety of black box optimisation problems outside of deep learning.

Need for More Cell Tracking Data

The lack of diverse, annotated cell tracking data is a major limitation in the field. Projects such as the Protein Data Bank have set a precedent for the successful standardisation and sharing of data, which has been a key driver in the development of highly accurate deep learning models such as AlphaFold. A similar data bank is being set up, where cell microscopy time lapses are accompanied by the specific imaging setup, cell type, conditions, etc., while allowing flexibility in uploading proposed ground truth annotations, as these may be specific to the problem being assessed by researchers and may differ between subjective human annotators. This could provide researchers worldwide with time-lapse cell microscopy images that can be used for a variety of purposes, and most importantly for cell tracking, it could be used to build a general ML-based model for cell tracking (and segmentation) that can handle diverse datasets.

Essential to this hypothetical scenario is overcoming the current lack of standardised frameworks for annotating and storing tracking data, which can lead to inconsistencies and challenges in data integration and comparative analysis across studies (56). A community-wide effort is needed to develop a standardised format for storing cell tracking data that is flexible enough to accommodate different datasets and tracking algorithms.

There are currently several open-source time-lapse cell tracking datasets available, such as the Cell Tracking Challenge dataset (57), which we used in our study, and the DeepCell dataset (58). However, these datasets are limited in size and diversity, and there is a need for more comprehensive datasets covering a wider range of cell types, imaging conditions and biological processes.

Limited Timelapse Cell Microscopy Data A major limitation of this study is the lack of available ground truth annotated time-lapse microscopy data. This is a major problem in the field of cell tracking, as making ground truth annotations is very laborious, there is little effort to standardise cell tracking annotations, and there is little incentive to publish ground truth annotated cell tracking datasets, as they are usually very specific to the problem being evaluated by the researcher. These protein structures have been subject to community scrutiny and standardisation, allowing them to be used as training data by deep learning models such as AlphaFold.

It would be highly beneficial to establish a similar framework/database for ground truth cell tracking annotations. These time lapses could be accompanied by the specific imaging setup, cell type, conditions, etc., and could provide researchers worldwide with cell microscopy time lapses that can be used for a variety of purposes, and most importantly for cell tracking, it could be used to build a general ML-based model for cell tracking (and segmentation) that can handle diverse datasets.

AOGM Weights No experiments were performed to determine the optimal weights for the AOGM metric. For example, AOGM-A is a modification of the AOGM that only takes into account edges to calculate the metric where: $w_{NS} = w_{FN} = w_{FP} = 0$ and

$w_{ED}, w_{EA}, w_{EC} > 0$ (59), it could be better able to capture a cell tracking algorithm's association skills. Future research could explore how different weight settings in the AOGM metric affect the efficiency of the optimisation process.

Need for Improved Annotation Tools For our implementation to be useful to cell biology researchers, it is essential to have a subset of your data annotated. This can become a significant bottleneck, due to the limited availability of user-friendly tools for annotating cell tracking data.

While Napari (60) can be used to annotate cell tracking data, it still requires users to annotate their data from scratch. It could also benefit from improvements to its user interface to make it more user-friendly for cell tracking purposes.

DeepCell Label, an easy-to-use online annotation software (61), is a step in the right direction by providing a user-friendly interface for annotating cell tracking data. However, it is limited in scope and could be improved by incorporating more advanced features such as automatic track generation and correction.

An effective cell tracking annotation tool could take automatically generated tracks as a starting point and allow users to refine them by marking correct connections between cells and adjusting incorrect ones. Such software could use Bayesian inference or a reinforcement learning framework to learn from user corrections and automatically adjust similar tracks across the dataset. The interface could iteratively present tracks for user approval or correction, possibly showing the user a set of likely alternatives for easy selection, and continue this process until the entire dataset is annotated or the rolling error rate falls below a specified threshold.

Conclusion

In this study, we developed a novel framework for optimising cell tracking software using Bayesian optimisation. Our results showed that Bayesian optimisation can outperform random search and grid search in terms of tracking accuracy and mitotic branching correctness. Our study highlights the potential of Bayesian optimisation as a powerful tool for optimising cell tracking software and improving tracking accuracy. We also identified several key considerations for those intending to use our software or construct their own parameter optimisation framework for cell tracking software. Future work should focus on validating the optimised parameters on separate datasets and exploring the impact of parameter optimisation techniques on the wider biological community. In addition, there is a need for more diverse and annotated cell tracking data to facilitate the development of more accurate and robust cell tracking algorithms.

EXPERIMENTAL PROCEDURES

Data Acquisition

There are 10 2D and 10 3D datasets in the Cell Tracking Challenge dataset (15). Several datasets were excluded from our analysis. *Fluo-C2DL-Huh7* was excluded due to a lack of reference segmentations. Due to computational constraints, we set the maximum file size to 5GB, excluding *Fluo-C3DH-H157*, *Fluo-N3DL-DRO*, *Fluo-N3DL-TRIC*, and *Fluo-N3DL-TRIF*.

Dataset	Description
2D Datasets	
BF-C2DL-HSC	Mouse hematopoietic stem cells.
BF-C2DL-MuSC	Mouse muscle stem cells.
DIC-C2DH-HeLa	HeLa cells.
Fluo-C2DL-MSC	Rat mesenchymal stem cells.
Fluo-N2DH-GOWT1	GFP-GOWT1 mouse stem cells.
Fluo-N2DL-HeLa	HeLa cells stably expressing H2b-GFP.
PhC-C2DH-U373	Glioblastoma-astrocytoma U373.
PhC-C2DL-PSC	Pancreatic stem cells.
Fluo-N2DH-SIM+	Simulated nuclei of HL60 cells.
3D Datasets	
Fluo-C3DH-A549	GFP-actin-stained A549 Lung Cancer cells.
Fluo-C3DL-MDA231	MDA231 human breast carcinoma cells.
Fluo-N3DH-CE	*C. elegans* developing embryo.
Fluo-N3DH-CHO	Chinese Hamster Ovarian (CHO) nuclei.
Fluo-N3DL-DRO	Developing *Drosophila melanogaster* embryo.
Fluo-C3DH-A549-SIM	Simulated GFP-actin-stained A549 Lung Cancer cells.
Fluo-N3DH-SIM+	Simulated nuclei of HL60 cells.

Table 1: 2D and 3D Datasets and their descriptions.

Optimisation loop

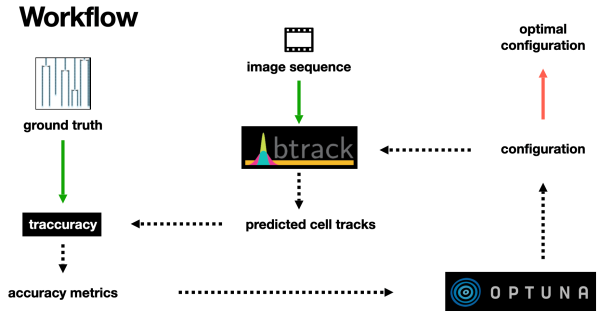

Figure 8: Schematic of the workflow for the optimisation of the btrack parameters. The workflow consists of three main steps: (1) Optuna suggesting a parameter configuration (2) cell tracking with said configuration, and (3) evaluation of tracking using traccuracy (62), these metrics are then fed back into Optuna, closing the loop. When a set number of trials has been performed the loop halts and the configuration that resulted in the best outcome is saved.

Acyclic Oriented Graphs Matching (AOGM) measure

The Acyclic Oriented Graphs Matching (AOGM) measure, as described by Matula et al. (59), quantifies the accuracy of a computed graph $G_C = (V_C, E_C)$ relative to a reference graph $G_R = (V_R, E_R)$. This metric considers various aspects of graph structure, from vertex accuracy to edge semantics, providing a comprehensive assessment of algorithm performance. We calculated the AOGM from the ground truths provided by the Cell Tracking Challenge using the `traccuracy` Python package.

Vertex Classification Vertices are categorised based on detection outcomes:

- **True Positives (TP)**: Reference vertices correctly detected and uniquely paired with a computed vertex.
- **False Negatives (FN)**: Reference vertices not detected in the computed graph.
- **False Positives (FP)**: Computed vertices without a corresponding reference vertex.
- **Non-Split Vertices (NS)**: Computed vertices that should have been split to match multiple reference vertices.

Edge Classification Edges are analysed as follows:

- **Redundant Edges (ED)**: Extra edges in the computed graph not found in the reference graph.
- **Missing Edges (EA)**: Edges in the reference graph but absent in the computed graph.
- **Edges with Wrong Semantics (EC)**: Edges present in both graphs but differing in connectivity or semantics.

AOGM Calculation The cost of transforming G_C into G_R is calculated as:

$$\begin{aligned}\text{AOGM} = w_{NS} \cdot NS + w_{FN} \cdot FN + w_{FP} \cdot FP \\ + w_{ED} \cdot ED + w_{EA} \cdot EA + w_{EC} \cdot EC \end{aligned} \quad (1)$$

Here, $w_{NS}, w_{FN}, w_{FP}, w_{ED}, w_{EA}$, and w_{EC} are the weights assigned to each operation type. For our experiments, all weights were set to 1.

Mitotic Branching Correctness (MBC)

MBC, as outlined by Bise et al. (63), specifically measures the accuracy of identifying mother-daughter relationships during mitosis. It is quantified as the ratio of correctly identified mitotic branches to the total number of mitotic events observed 2.

$$\text{MBC} = \frac{\text{Number of correctly detected mitotic branches}}{\text{Total number of mitotic events}} \quad (2)$$

A mitotic event is considered correctly identified if:

- The tracked cell i' at time t' correctly corresponds to the actual cell i at time t.
- The daughters j' and k' of the tracked cell i' correspond to the actual daughters j and k of cell i.
- The time difference $\epsilon = \|t - t'\|$ between the tracked and actual mitotic events is below a defined threshold θ_ϵ which is commonly set to 10.

Tree Visualisation

To construct Figure 7, we developed a tree visualisation tool based on the source code of the napari arboretum plugin (64). This tool allows the visualisation of lineage trees outside of the napari GUI, providing greater flexibility and functionality.

Software and Libraries

In this study, we used several Python packages for data analysis and cell tracking. Cell tracking was performed using `btrack` (version 0.6.4). Data manipulation and analysis was performed using `numpy` (version 1.26.4) and `pandas` (version 2.2.1). Job management was facilitated by `joblib` (version 1.3.2). Hyperparameter optimisation was performed using `optuna` (version 3.5.0). In addition, tracking accuracy and performance metrics were evaluated using `traccuracy` (version 0.1.0).

Hardware

Experiments were conducted on a MacBook Air M2 with 16GB of RAM, an 8-core M2 CPU, and an 8-core GPU. The software has been tested on MacOS 14.5 and found to be adequate.

Code Availability

Comprehensive documentation, user-friendly tutorials, examples and installation instructions will be available on GitHub from 28 May. These resources are designed to facilitate reproducibility and ease of use. The code is available in the following repositories:

- https://github.com/quantumjot/btrack/tree/main/examples
- https://github.com/timsmsmsm/rl-tracking

Acknowledgements: My sincere thanks to Dr. Alan R. Lowe for his guidance and genuine care and support throughout this project. My thanks also go out to my dear friends Anna Maria Papaoikonomou and Ado Farsi, who have been a great source of inspiration and joy during this difficult time.

Word Count: 4979

Author Contributions: Alan R. Lowe and Tim August B. Huygelen conceived the major research ideas and developed the theoretical framework. Tim August B. Huygelen designed the experimental setup, developed the methodology, performed the experiments, and collected the data. He also analysed the data, interpreted the results, wrote the manuscript and prepared the first draft. In addition, Tim August B. Huygelen revised the manuscript for important intellectual content, developed the software, and performed the computational analyses. Alan R. Lowe supervised the research, provided technical assistance, and managed the project.

REFERENCES

1. Anna Bove, Daniel Gradeci, Yasuyuki Fujita, Shiladitya Banerjee, Guillaume Charras, and Alan R Lowe. Local cellular neighborhood controls proliferation in cell competition. *Molecular biology of the cell*, 28(23):3215–3228, 2017.

2. Kristina Ulicna, Giulia Vallardi, Guillaume Charras, and Alan R Lowe. Automated deep lineage tree analysis using a bayesian single cell tracking approach. *Frontiers in Computer Science*, 3:734559, 2021.

3. Robert Hooke. *Micrographia: or some Physiological Descriptions of Minute Bodies Made by Magnifying Glasses*. Royal Society, 1665. Digitally accessed from Wikisource on 21-May-2024.

4. Anthoni van Leeuwenhoek. *Alle de brieven. Deel 1: 1673-1676*. N.V. Swets & Zeitlinger, Amsterdam, 1939. Contains "Letter from Leeuwenhoek to Oldenburg, 7 September 1674." Digitally accessed from DBNL (KB, nationale bibliotheek) on 21 May 2024.

5. Nick Lane. The unseen world: reflections on leeuwenhoek (1677)'concerning little animals'. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 370(1666):20140344, 2015.

6. Demorest Davenport, Glenn J. Culler, John O. B. Greaves, Richard B. Forward, and William G. Hand. The investigation of the behavior of microorganisms by computerized television. *IEEE Transactions on Biomedical Engineering*, BME-17(3):230–237, 1970.

7. Anne E Carpenter, Thouis R Jones, Michael R Lamprecht, Colin Clarke, In Han Kang, Ola Friman, David A Guertin, Joo Han Chang, Robert A Lindquist, Jason Moffat, et al. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome biology*, 7:1–11, 2006.

8. Valeria Levi and Enrico Gratton. Exploring dynamics in living cells by tracking single particles. *Cell biochemistry and biophysics*, 48:1–15, 2007.

9. Erik Meijering, Oleh Dzyubachyk, Ihor Smal, and Wiggert A van Cappellen. Tracking in cell and developmental biology. In *Seminars in cell & developmental biology*, volume 20, pages 894–902. Elsevier, 2009.

10. Vladimír Ulman et al. An objective comparison of cell-tracking algorithms. *Nature methods*, 14(12):1141–1152, 2017.

11. Neda Emami, Zahra Sedaei, and Reza Ferdousi. Computerized cell tracking: Current methods, tools and challenges. *Visual Informatics*, 5(1):1–13, 2021.

12. Takanobu A Katoh, Yohsuke T Fukai, and Tomoki Ishibashi. Optical microscopic imaging, manipulation, and analysis methods for morphogenesis research. *Microscopy*, page dfad059, 2023.

13. Dominik Schienstock and Scott N. Mueller. Moving beyond velocity: Opportunities and challenges to quantify immune cell behavior. *Immunological Reviews*, 306(1):123–136, 2022.

14. Tal Ben-Haim and Tammy Riklin Raviv. Graph neural network for cell tracking in microscopy videos. In *European Conference on Computer Vision*, pages 610–626. Springer, 2022.

15. Martin Maška, Vladimír Ulman, Pablo Delgado-Rodriguez, Estibaliz Gómez-de Mariscal, Tereza Nečasová, Fidel A Guerrero Peña, Tsang Ing Ren, Elliot M Meyerowitz, Tim Scherr, Katharina Löffler, et al. The cell tracking challenge: 10 years of objective benchmarking. *Nature Methods*, 20(7):1010–1020, 2023.

16. Tim Scherr, Katharina Löffler, Moritz Böhland, and Ralf Mikut. Cell segmentation and tracking using cnn-based distance predictions and a graph-based matching strategy. *PLoS One*, 15(12):e0243219, 2020.

17. Katharina Löffler and Ralf Mikut. Embedtrack—simultaneous cell segmentation and tracking through learning offsets and clustering bandwidths. *IEEE Access*, 10:77147–77157, 2022.

18. Klas E. G. Magnusson et al. Global linking of cell tracks using the viterbi algorithm. *IEEE Transactions on Medical Imaging*, 34(4):911–929, 2015.

19. Ko Sugawara, Çağrı Çevrim, and Michalis Averof. Tracking cell lineages in 3d by incremental deep learning. *Elife*, 11:e69380, 2022.

20. Christopher J Soelistyo, Kristina Ulicna, and Alan R Lowe. Machine learning enhanced cell tracking. *Frontiers in Bioinformatics*, 3, 2023.

21. K. Löffler, T. Scherr, and R. Mikut. A graph-based cell tracking algorithm with few manually tunable parameters and automated segmentation error correction. *PLOS ONE*, 16(9):e0249257, 2021.

22. Oliver J Meacock and William M Durham. Tracking bacteria at high density with fast, the feature-assisted segmenter/tracker. *PLOS Computational Biology*, 19(10):e1011524, 2023.

23. Nicolas Chenouard, Ihor Smal, Fabrice De Chaumont, Martin Maška, Ivo F Sbalzarini, Yuanhao Gong, Janick Cardinale, Craig Carthel, Stefano Coraluppi, Mark Winter, et al. Objective comparison of particle tracking methods. *Nature methods*, 11(3):281–289, 2014.

24. Nao Nishida-Aoki and Taranjit S Gujral. Emerging approaches to study cell–cell interactions in tumor microenvironment. *Oncotarget*, 10(7):785, 2019.

25. Jungmok Seo et al. High-throughput approaches for screening and analysis of cell behaviors. *Biomaterials*, 153:85–101, 2018.

26. Yupeng Hong, Francia Fang, and Qi Zhang. Circulating tumor cell clusters: What we know and what we expect. *International journal of oncology*, 49(6):2206–2216, 2016.

27. Sam H. Au et al. Clusters of circulating tumor cells: a biophysical and technological perspective. *Current opinion in biomedical engineering*, 3:13–19, 2017.

28. Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019.

29. Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging {AI} applications. In *13th USENIX symposium on operating systems design and implementation (OSDI 18)*, pages 561–577, 2018.

30. Y. Fukai and K. Kawaguchi. Laptrack: Linear assignment particle tracking with tunable metrics. *bioRxiv*, pages 123–135, 2022.

31. L. Taveira, T. Kurc, A. Melo, J. Kong, E. Bremer, J. Saltz, et al. Multi-objective parameter auto-tuning for tissue image segmentation workflows. *Journal of Digital Imaging*, 32(3):521–533, 2018.

32. Duc Phu Chau, Julien Badie, François Brémond, and Monique Thonnat. Online tracking parameter adaptation based on evaluation. In *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 189–194. IEEE, 2013.

33. Yoshihiko Ozaki, Yuki Tanigaki, Shuhei Watanabe, Masahiro Nomura, and Masaki Onishi. Multiobjective tree-structured parzen estimator. *Journal of Artificial Intelligence Research*, 73:1209–1250, 2022.

34. G. Rong et al. Comparison of tree-structured parzen estimator optimization in three typical neural network models for landslide susceptibility assessment. *Remote Sensing*, 13(22):4694, 2021.

35. Z. Wang. Hces-net: Hepatic cystic echinococcosis classification ensemble model based on tree-structured parzen estimator and snap-shot approach. *Medical Physics*, 50(7):4244–4254, 2023.

36. Y. Wang and X. Ni. A xgboost risk model via feature selection and bayesian hyper-parameter optimization. *International Journal of Database Management Systems*, 11(01):01–17, 2019.

37. H. Nguyen, J. Liu, and E. Zio. A long-term prediction approach based on long short-term memory neural networks with automatic parameter optimization by tree-structured parzen estimator and applied to time-series data of npp steam generators. *Applied Soft Computing*, 89:106116, 2020.

38. Yang Xu, Weijun Gao, Fanyue Qian, and Yanxue Li. Potential analysis of the attention-based lstm model in ultra-short-term forecasting of building hvac energy consumption. *Frontiers in Energy Research*, 9:730640, 2021.

39. J. Liang et al. Intelligent fault diagnosis of rotating machinery using lightweight network with modified tree-structured parzen estimators. *IET Collaborative Intelligent Manufacturing*, 4(3):194–207, 2022.

40. Xinyi Zhang, Chengyuan Dai, Weiyu Li, and Yang Chen. Prediction of compressive strength of recycled aggregate concrete using machine learning and bayesian optimization methods. *Frontiers in Earth Science*, 11:1112105, 2023.

41. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

42. R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. Test problems for large-scale multiobjective and many-objective optimization. *IEEE Transactions on Cybernetics*, 47(12):4108–4121, 2017.

43. R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 20(5):773–791, 2016.

44. Rupam Kundu, Rohan Mukherjee, Shantanab Debchoudhury, Swagatam Das, Ponnuthurai N Suganthan, and Thanos Vasilakos. Improved cma-es with memory based directed individual generation for real parameter optimization. In *2013 IEEE Congress on Evolutionary Computation*, pages 748–755. IEEE, 2013.

45. Ilya Loshchilov, Marc Schoenauer, Michele Sebag, and Nikolaus Hansen. Maximum likelihood-based online adaptation of hyper-parameters in cma-es. In *International Conference on Parallel Problem Solving from Nature*, pages 70–79. Springer, 2014.

46. Konstantinos Varelas. Benchmarking large scale variants of cma-es and l-bfgs-b on the bbob-largescale testbed. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1937–1945, 2019.

47. Raymond Ros and Nikolaus Hansen. A simple modification in cma-es achieving linear time and space complexity. In *International conference on parallel problem solving from nature*, pages 296–305. Springer, 2008.

48. Nacim Belkhir, Johann Dréo, Pierre Savéant, and Marc Schoenauer. Per instance algorithm configuration of cma-es with limited budget. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 681–688, 2017.

49. Stefan Kern, Sibylle D Müller, Nikolaus Hansen, Dirk Büche, Jiri Ocenasek, and Petros Koumoutsakos. Learning probability distributions in continuous evolutionary algorithms–a comparative review. *Natural Computing*, 3:77–112, 2004.

50. Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. *ArXiv*, abs/1810.05270, 2018.

51. Petro Liashchynskyi and Pavlo Liashchynskyi. Grid search, random search, genetic algorithm: a big comparison for nas. *arXiv preprint arXiv:1912.06059*, 2019.

52. Xinghua Lou, Martin Schiegg, and Fred A Hamprecht. Active structured learning for cell tracking: algorithm, framework, and usability. *IEEE transactions on medical imaging*, 33(4):849–860, 2014.

53. Tobias Fleck and J. Marius Zoellner. Tuning multi object tracking systems using bayesian optimization. In *2021 IEEE 24th International Conference on Information Fusion (FUSION)*, pages 1–8, 2021.

54. Zi Wang et al. Hierarchical deep reinforcement learning reveals a modular mechanism of cell movement. *Nature machine intelligence*, 4(1):73–83, 2022.

55. Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural Information Processing Systems 33*, 2020.

56. Patryk Burek, Nico Scherf, and Heinrich Herre. Ontology patterns for the representation of quality changes of cells in time. *Journal of biomedical semantics*, 10:1–18, 2019.

57. Martin Maška, Vladimír Ulman, David Svoboda, Pavel Matula, Petr Matula, Cristina Ederra, Ainhoa Urbiola, Teresa España, Srinivasan Venkatesan, D. M. Waruna Balak, et al. A benchmark for comparison of cell tracking algorithms. *Bioinformatics*, 30(11):1609–1617, 2014.

58. David A Van Valen, Takamasa Kudo, Keara M Lane, Derek N Macklin, Nicolas T Quach, Mialy M DeFelice, Inbal Maayan, Yu Tanouchi, Euan A Ashley, and Markus W Covert. Deep learning automates the quantitative analysis of individual cells in live-cell imaging experiments. *PLoS computational biology*, 12(11):e1005177, 2016.

59. P. Matula, M. Maška, D. V. Sorokin, P. Matula, C. Ortiz-de Solórzano, and M. Kozubek. Cell tracking accuracy measurement based on comparison of acyclic oriented graphs. *PLoS ONE*, 10(12):e0144959, 2015.

60. Napari Contributors. napari: a multi-dimensional image viewer for python. *Zenodo https://doi.org/10.5281/zenodo*, 3555620, 2019.

61. Noah F Greenwald, Geneva Miller, Erick Moen, Alex Kong, Adam Kagel, Thomas Dougherty, Christine Camacho Fullaway, Brianna J McIntosh, Ke Xuan Leow, Morgan Sarah Schwartz, et al. Whole-cell segmentation of tissue images with human-level performance using large-scale data annotation and deep learning. *Nature biotechnology*, 40(4):555–565, 2022.

62. Janelia-Trackathon-2023. traccuracy: Utilities for computing common accuracy metrics on cell tracking challenge solutions with ground truth. https://github.com/Janelia-Trackathon-2023/traccuracy, 2023. GitHub repository.

63. Ryoma Bise, Zhaozheng Yin, and Takeo Kanade. Reliable cell tracking by global data association. In *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 1004–1010, 2011.

64. Alan R Lowe. napari-arboretum. https://github.com/lowe-lab-ucl/arboretum, 2023.

65. Alan R. Lowe. Bayesian tracker documentation website: Configuration, 2024. Accessed: 2024-05-20.

FOOTNOTES

Abbreviations:

- BO: Bayesian Optimization
- TPE: Tree-structured Parzen Estimator
- CMA-ES: Covariance Matrix Adaptation Evolution Strategy
- NSGA-II: Non-dominated Sorting Genetic Algorithm II
- GP: Gaussian Process
- KDE: Kernel Density Estimation
- EI: Expected Improvement
- UCB: Upper Confidence Bound
- AOGM: Acyclic Oriented Graphs Matching
- MBC: Mitotic Branching Correctness
- ML: Machine Learning
- DIC: Differential Interference Contrast
- BF: Bright Field
- PhC: Phase Contrast
- Fluo: Fluorescence

SUPPLEMENTARY MATERIAL

Parameter	Range	Type	Model	Description
theta_dist	$(0, 99.99)$	Continuous	Hypothesis Model	Threshold distance from the edge of the FOV to add an initialization or termination hypothesis.
lambda_time	$(0, 99.99)$	Continuous	Hypothesis Model	Scaling factor for the influence of time when determining initialization or termination hypotheses.
lambda_dist	$(0, 99.99)$	Continuous	Hypothesis Model	Scaling factor for the influence of distance at the border when determining initialization or termination hypotheses.
lambda_link	$(0, 99.99)$	Continuous	Hypothesis Model	Scaling factor for the influence of track-to-track distance on linking probability.
lambda_branch	$(0, 99.99)$	Continuous	Hypothesis Model	Scaling factor for the influence of cell state and position on division (mitosis/branching) probability.
theta_time	$(0, 99.99)$	Continuous	Hypothesis Model	Threshold time from the beginning or end of movie to add an initialization or termination hypothesis.
dist_thresh	$(0, 99.99)$	Continuous	Hypothesis Model	Isotropic spatial bin size for considering hypotheses.
time_thresh	$(0, 99.99)$	Continuous	Hypothesis Model	Temporal bin size for considering hypotheses.
apop_thresh	$(0, 99)$	Integer	Hypothesis Model	Number of apoptotic detections to be considered a real apoptosis.
segmentation_miss_rate	$(0, 1.0)$	Continuous	Hypothesis Model	Miss rate for the segmentation, e.g., 0.01 for 1/100 segmentations incorrect.
p_sigma	$(0, 500)$	Continuous	Motion Model	Simplified estimated error in process. Used to calculate Q using $Q = G^T G$.
g_sigma	$(0, 500)$	Continuous	Motion Model	Simplified estimated error in process. Used to calculate Q using $Q = G^T G$.
r_sigma	$(0, 500)$	Continuous	Motion Model	Estimated error in measurements.
accuracy	$(0.1, 10)$	Continuous	Miscellaneous	Integration limits for calculating the probabilities.
max_lost	$(1, 10)$	Integer	Motion Model	Number of frames without observation before marking as lost.
prob_not_assign	$(0.0, 1.0)$	Continuous	Motion Model	The default probability to not assign a track.
max_search_radius	$(0, 1000)$	Integer	Miscellaneous	Maximum search radius for the tracking algorithm in isotropic unit of the data.
div_hypothesis	$(0, 1)$	Integer (Boolean)	Hypothesis Model	Enables/disables division hypotheses, when set to 0 no mitotic events will be detected.

Table 2: Parameter ranges, types, associated models, and descriptions. Parameter descriptions are from the *btrack* documentation website (65).