SUPPORTING MATERIAL

POTATO: Automated pipeline for batch analysis of optical tweezers data

Stefan Buck [†1], Lukas Pekarek[†1], Neva Caliskan* [1,2]

[†] Authors contributed equally to this work.

* Corresponding author

[1] Helmholtz Institute for RNA-based Infection Research (HIRI), Würzburg, Germany

[2] Medical Faculty, Julius-Maximilians University Würzburg, Würzburg, Germany

Script structure

The script is written in Python 3 and split into multiple parts for clarity. The first part, "POTATO_GUI", defines the GUI with all necessary functions and input variables. When the GUI is started, the default values of the input variables are loaded from the "POTATO_config" file. The GUI was created and structured using the standard Tkinter python package. A parallel subprocess initiates from this main process when a folder is selected for force ramp analysis to perform computationally demanding data-processing. This way the GUI remains responsive during computation. All the functions used for data preprocessing and step recognition are defined in the "POTATO_preprocessing" and the "POTATO_find_steps" files respectively. The functions used for curve fitting are defined in another file, "POTATO_fitting". For computation, we mainly use matplotlib and NumPy packages, as well as the lumicks.pylake package for fitting (**Table S4**). The subprocess is a daemon process spawned by the main process and therefore stops as soon as the GUI terminates the main process. The last part, "POTATO_constantF", is executed by the main thread as it only analyzes one constant force file at a time. The results are exported in different CSV files or as PNG images.

Graphical user interface

We designed a graphical user interface (GUI) that allows users to easily adjust the analysis steps and parameters according to their needs and select between three different input data formats. This enables the GUI to load data from every OT instrument. The GUI is separated into multiple tabs, resulting in easy and intuitive navigation without overloading the individual windows. The "POTATO_config" file, included in the POTATO repository, contains the default parameters, which are loaded into the GUI. The most commonly changed parameters can be found in the first tab, “Folder Analysis”, so a basic analysis can be performed right away (press enter to confirm changed parameters). Alternatively, before each analysis, all parameters can be adjusted in the 'Advanced Settings' tab to suit the data set. In addition, we implemented the possibility to selectively export results. Each analysis creates a new folder with a timestamp directly in the analyzed directory. The used parameters are exported as well so that parameters can be optimized later. The second tab, “Show Single File”, provides a control mechanism for data preprocessing. A single file can be loaded, and the unfiltered data are plotted together with the filtered data, which streamlines troubleshooting. Finally, there is a third tab for “Constant Force Analysis”.

Input data

The presented pipeline accepts three different input data formats. Two of them are based on the default hdf5 output format of Lumicks C-Trap – one is predefined for high-frequency data (using the piezo-tracking function of the instrument), and the second is for low-frequency data (using video recognition). The third data format is a basic CSV file format with force and distance values in the first and second columns. Force data need to be in [pN], whereas the unit of distance data can be specified either as [µm] or [nm] in the GUI. Thus, our pipeline can process force-distance data from virtually any optical tweezers machine. In addition, entire directories containing force-ramp data files can be selected and processed simultaneously.

Data output

Depending on individual analysis requirements, different export settings can be selected. The down sampled and filtered data are exported in CSV format (smooth) for each file by default. The identified (un)folding steps by derivatives of force and distance are exported together with the steps identified by both strategies (common steps) into a single CSV file. All identified steps of all curves in the analyzed folder are gathered into a single results file for quantitative analysis. The respective summary figure containing the plot of preprocessed data, trimmed data, and both derivatives with

marked steps is exported. The plots of fitted data, together with the fitting parameters and model data, are exported as PNG and CSV files, respectively.

Artificial data generation

To test the limits of the algorithm, artificial data with a single step per curve were generated. The fully folded part of force-distance curves was modeled using an equation for extensible WLC models (**Eq. 4**). The partially unfolded region was modeled using a combination of WLC and FJC models (**Eq. 5 and 6**). The force value at which the step occurs, the contour length change between the unfolded and folded region, and the drop in force during the step, are the parameters for data generation. The first parameter was set to occur between 10-40 pN with a 5 pN resolution. The curves were generated with a contour length change from 1-40 nm with a 1 nm resolution and a force drop of 1-5 pN with a 0.5 pN resolution. To mimic the (Gaussian) noise affecting the raw data, we employed the NumPy random normal distribution function (1).

Since the (un)folding step is generally defined as a drop in force (one of the parameters) and a sudden increase in distance (not a parameter), the data generated by this script also contained combinations that did not increase distance. We used these curves showing no increase in the distance as negative controls.

Augmentation of low-frequency data

During analysis of the freely available data from Neupane and Zhao et al., 2021, we had to employ the data augmentation approach to increase the precision of the analysis. For the best output, ideally raw data should be directly curated in POTATO and at least >2000 data points are available. The augmentation was performed as follows. For each two consecutive data points in the original data, we divided the linear space between them by factor of 100 to get positions for new data points. Starting from the first original data point, we consecutively added 99 new data values always increasing by the previously calculated increment +/- randomly assigned noise in force and distance dimensions using random gauss function with the parameters mu=0 and sigma=0.5. The newly created files where then analyzed as csv files by POTATO.

Manual data analysis

POTATO GUI also contains a tab that provides the user with the option to manually mark steps, fit models and calculate the work for FD curves (Manual Analysis – TOMATO). Manual analysis is particularly useful to evaluate the precision of the automated analysis and perform parameter optimization. Furthermore, manual analysis is a convenient option for the analysis of FD curves which cannot be analyzed properly by the batch analysis. To speed up the manual analysis we implemented several keyboard shortcuts, which allow switching between FD curves under a given directory and marking steps with save, delete functions among others. For the manual analysis, the initial step is to mark the beginning and end of each (un)folding event. Afterward, similar to the batch analysis, different parts of the curves will be fitted automatically based on the parameters entered at the GUI. In addition, the work corresponding to each (un)folding event is calculated based on the fitting (**Eq. 8**). During manual analysis, certain parameter constraints and fitting parameters such as contour length, persistence length, stiffness, distance and force offsets can be defined. A detailed set of parameter constraints can also be found under the 'Advanced Settings' tab. For further description, we suggest the reader to refer to the readme file on Github (https://github.com/REMI-HIRI/POTATO).

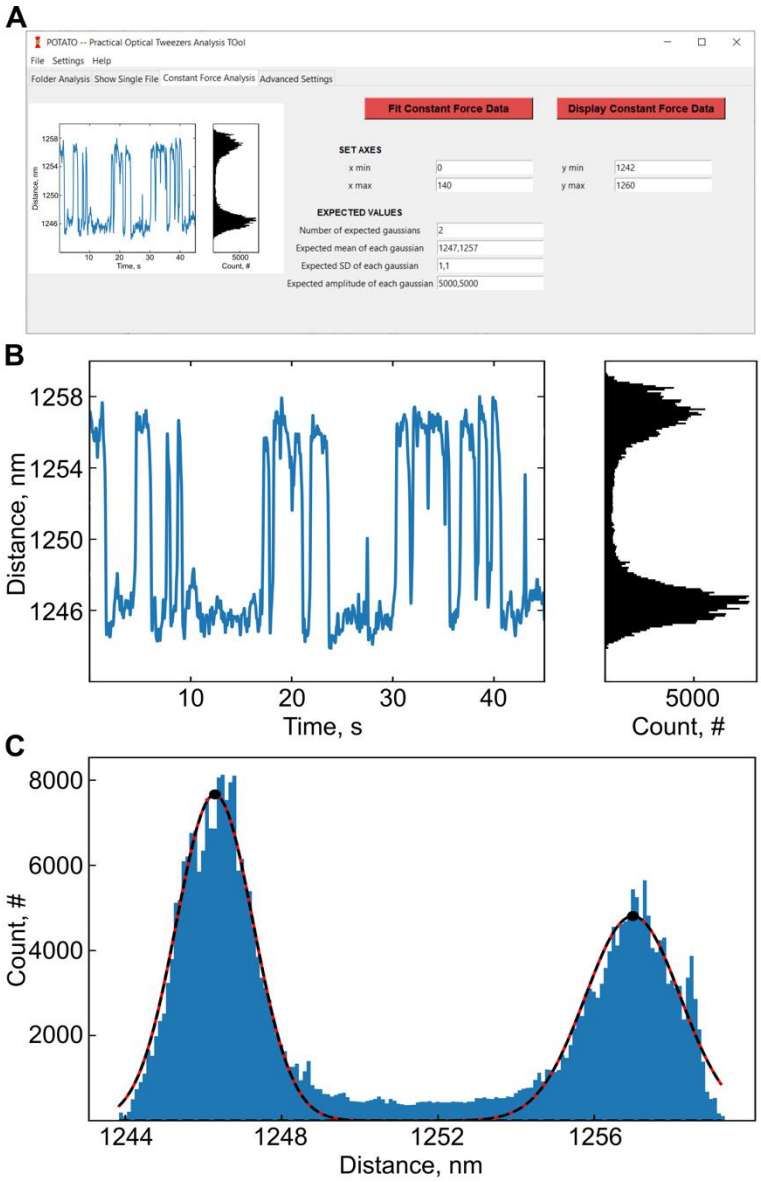

FIGURE S1: Constant force data analysis in POTATO: (A) GUI tab containing the constant force analysis features, **(B)** Display constant force data output; (left) distance over time plot, (right) histogram of the distance over time values. **(C)** Fit constant force data output showing the histogram of distance values distributions and the two gaussian functions fitted.

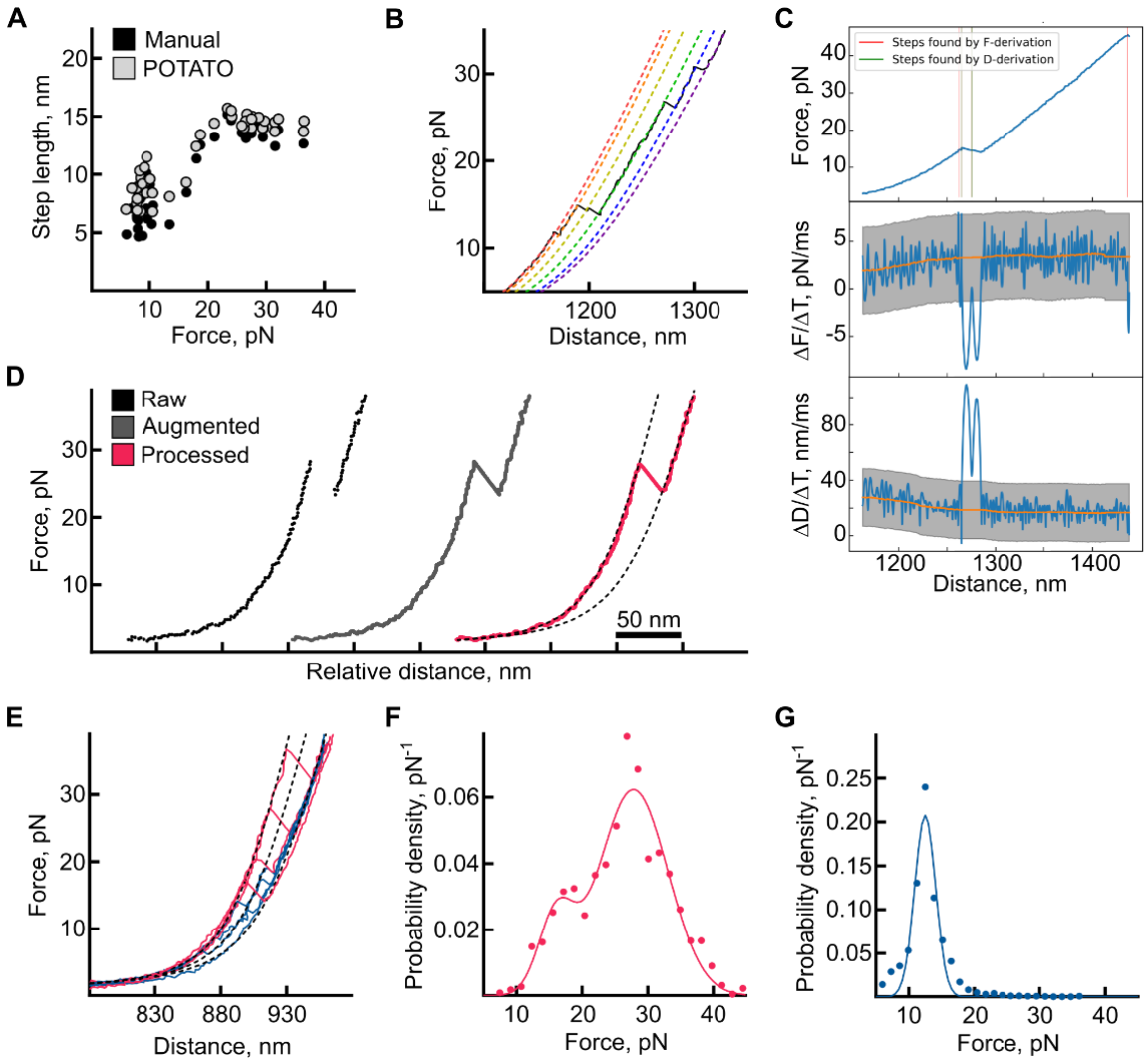

FIGURE S2: Analysis of experimental data by POTATO. **(A)** Comparison of unfolding events marked in a subset of the data analysed manually (black) or with POTATO (grey). (**B**) Example FD curve (black, solid) with five unfolding steps fitted by POTATO (colored, dashed). **(C)** Example analysis output from POTATO showing the trimmed FD curve (up), force derivative data (middle), and distance derivative data (bottom). An intermediate conformer is detected by POTATO during the unfolding. Other FD curves confirmed the presence of an even more stable and distinct intermediate step. (**D-G**) Analysis of experimental data published in Neupane and Zhao et al., 2021 (subset with 6nt spacer) using POTATO; (**D**) Comparison of raw data (black), data after augmentation (see also supplementary methods, grey), and data processed by POTATO (pink). **(E)** Example unfolding (pink) and refolding (blue) FD curves (n=4). (**F**) Distribution of unfolding forces for unfolding curves with single unfolding step (N=1378). (**G**) Refolding force distribution for all refolding curves (N=1861) shows a single peak around 12 pN with refolding steps detected at forces as low as 6 pN. (Un)folding distributions were overall similar to the analysis performed by Neupane and Zhao et al., 2021.

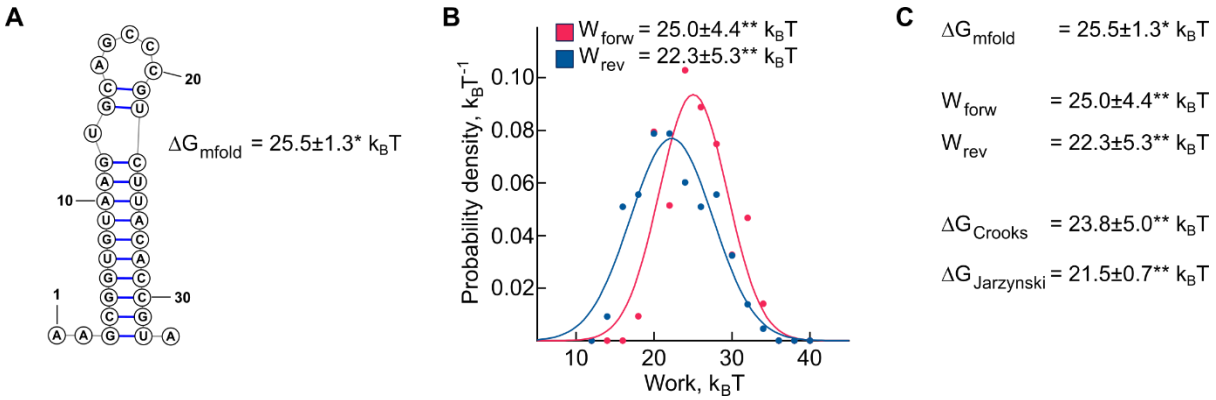

FIGURE S3: Extracting energy information from the experimental data. **(A)** Mfold predicted secondary structure of a simple hairpin of 30 nucleotides in lenght. **(B)** Distributions of measured work values for the unfolding (red) and refolding (blue) FD curves. **(C)** Energy and work values as predicted by Mfold (ΔG_{mfold}), measured (W_{forw} and W_{rev}) or calculated using Crooks Theorem (ΔG_{Crooks}) and Jarzynski equality ($\Delta G_{Jarzynski}$). * 5% standard error, **standard deviation.

TABLE S1: Parameters used throughout the pipeline and a short description.

Parameter	Description
Preprocessing	
Downsampling rate	Only every n^{th} value is taken for analysis, speeds up subsequent processing.
Butterworth filter degree	Defines the stringency of the filter.
Cut-off frequency	Signals with a frequency above this threshold are suppressed.
Force threshold, pN	Values lower than the threshold are excluded from the analysis.
Derivative	
Step d	Characterizes the interval between two values used for numerical derivative calculation.
Data frequency, Hz	The frequency at which data is recorded.
Statistics	
z-score	The number of standard deviation used to determine whether a given value is part of a normal distribution.
Moving median window size	The number of values considered for each median calculation.
SD difference threshold	Statistical analysis and data sorting are iterated until the difference between two consecutive SDs is below this value.
Fitting	
dsLp, nm	Persistence length of the double-stranded (folded) part of the tethered construct.
dsLc, nm	Contour length of double-stranded (folded) part of the tethered construct.
dsK0, pN	Stretch modulus of double-stranded (folded) part of the tethered construct.
Force offset, pN	Force offset of a given dataset; compensates for a shift in the dataset.
Distance offset, nm	Distance offset of a given dataset; compensates for a shift in the dataset.
ssLp, nm	Persistence length of the single-stranded (unfolded) part of the tethered construct.
ssLc, nm	Contour length of single-stranded (unfolded) part of the tethered construct.
ssK0, pN	Stretch modulus of single-stranded (unfolded) part of the tethered construct.

TABLE S2: Dependence of the performance measures on the z-score. Analysis of 2520 simulated data curves with steps occurring between 10-40 pN with different z-score values.

z-score / Parameter	3.2	3	2.7	2.5
True positives	1206	1267	1280	1303
True negatives	1101	1076	1073	1056
False positives	32	57	60	77
False negatives	181	120	107	84
Accuracy	0.915	0.930	0.934	0.936
Precision	0.974	0.957	0.955	0.944
Recall	0.870	0.913	0.923	0.939
Specificity	0.972	0.950	0.947	0.932
F1-Score	0.919	0.935	0.939	0.942

TABLE S3: Application of POTATO on experimental data generated from a simple hairpin and SARS-CoV-2 pseudoknot.

	Expected ΔLc, nm	Observed ΔLc, nm	Observed ΔLc, nm (Neupane et al. 2021)
Simple hairpin (30 nt)	17.7	16.4±2.8	-
SARS-CoV-2 frameshift pseudoknot (6 nt spacer)	34.7-36.3	34.8±2.0	35.6±0.4

TABLE S4: Python packages used in POTATO. Standard packages are not included in the table.

Package name	Link
h5py	https://www.h5py.org (2)
Pandas	https://pandas.pydata.org (3)
Scipy	https://www.scipy.org (4)
Matplotlib	https://matplotlib.org (5)
Lumicks.pylake	https://lumicks-pylake.readthedocs.io

SUPPORTING REFERENCES

1. Harris, C. R., K. J. Millman, . . . T. E. Oliphant. Array programming with NumPy. Nature 2020 585(7825):357-362, doi: 10.1038/s41586-020-2649-2.
2. Collette, A. Python and HDF5. O'Reilly; 2013
3. McKinney, W. Data Structures for Statistical Computing in Python. Proc. of the 9th Python in Science Conf. 2010; 445:51-56. doi: 10.25080/Majora-92bf1922-00a
4. Virtanen, P., R. Gommers, . . . Y. Vázquez-Baeza. 2020. SciPy 1.0: fundamental algorithms for scientific computing in python. Nat. Methods 2020; 17(3):261-272, doi: 10.1038/s41592-019-0686-2.
5. Hunter, J. D. 2007. Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering. 9(3):90-95, doi: 10.1109/MCSE.2007.55.