# Performance and Usability Improvements to an Image-Analysis Program for Microtubules

Justin Ragatz

Faculty Sponsors: Samantha Foley, Computer Science and Taviare L. Hawkins, Physics

## ABSTRACT

Microtubules must be rigid enough to support the structure of a cell, while simultaneously being dynamic enough for intracellular transport. Any abnormalities in this balance may result in cancer, birth defects, or cell death. MT-Flex is an image-analysis program used to study the persistence length of microtubules. The persistence length is a property that quantifies the stiffness of a polymer, such as a microtubule. However, the original design of the program took an excessive amount of time to run, did not include any automated error checking, and returned results in an unorganized format. Moreover, flaws in the design of the code made the application difficult to use. Upon removing all unnecessary reading of data and restructuring the code, a speedup of 2.019 was achieved. Furthermore, the structure of the code and format of the output is now more intuitive, making it easier to use for both experienced and first-time users.

## INTRODUCTION

The cytoskeleton is a series of intracellular proteins that supports a cell's shape and is essential for many cellular functions. This intracellular matrix is comprised of three protein filaments: actin filaments, intermediate filaments, and microtubules.[1] Unlike actin and intermediate filaments, microtubules are in a constant state of assembly and disassembly. This unique characteristic is essential for cell division, neuronal cell development, cellular maintenance, and ciliary beating in the lungs, kidneys, and intestines. Any abnormalities in the structure of a microtubule may result in cancer, birth defects, or cell death.[1] Thus, increasing our understanding of this process is of particular importance.

Microtubules must be rigid enough to support the structure of a cell, while simultaneously being dynamic enough to allow intracellular transport.[2] The flexibility of a polymer is measured by the persistence length of the object.[3] An object that is shorter than its persistence length is considered more flexible, while an object that is longer than its persistence length is considered more rigid. Studying this property begins with imaging a microtubule under fluorescence. These images, 1,000 in total, are read in by the MT-Flex application which divides each frame into 23-35 segments of equal length. For each segment the position, length, and angle are determined. These properties are then used to calculate the Fourier mode amplitudes for the first 25 modes. The resulting Fourier series approximates the shape of the microtubule. Given that the Fourier series is an approximation of the molecule's shape, the persistence length is of course inversely proportional to the variance of the mode amplitudes.[3] Thus, the persistence length of the molecule is determined using the first 2-4 modes of the Fourier series.

This paper describes the original MT-Flex code, the motivation for the changes made, and how these modifications improve the usability of the application. The original application section describes the purpose of the application, how the MT-Flex code is used, and the issues which motivated the changes to the application. The requirements section details the changes that were identified as essential to improving the usability of the application. The new design section summarizes the changes that were made to the application and the performance

improvements that were achieved. Finally, the paper concludes with a summary of how the modifications made benefit the scientists who use the application.

## ORIGINAL APPLICATION

The purpose of the MT-Flex code is to efficiently analyze the persistence length of a microtubule from a series of image files. Scientists are able to evaluate more data sets in less time by using this application to measure the microtubules, as opposed to doing this manually. The application is written in MATLAB due to the language's reputation for excellent matrix operation performance, and its popularity among science communities.

The first thing a scientist must do to use the application is to insert 1000 tiff files (for example *image1.tiff, image2.tiff, ... , image1000.tiff*) into the folder containing the MATLAB files. These files should all have the same prefix (for example "image") and this prefix must be placed into the *Movie_No.txt* file. The application reads in any tiff files that match the naming and numbering pattern. Next, the scientist must open the *MAIN_PROGRAM_MOBILIZED.m* and *MT_Final_Results.m* files. The camera magnification value located in these files must be updated with the level of magnification that was used in the microtubule imaging process. Additionally, the scientist must open the *Image_Stack_MT_Skeletonization.m* and *MT_Skeletonization.m* files to update the spur number and threshold value, respectively. The spur number limits the length of any branches on the microtubule image, and the threshold number defines the threshold for binary image conversion. Finally, the scientist must return to the *MAIN_PROGRAM_MOBILIZED.m* file and run the application. A summary of how to set up a run in list format is described in **Table1**.

**Table 1.** Summary of how to set up a run.

| Step | Task |
|---|---|
| 1 | Place tiff files in the main level of the working directory. |
| 2 | Enter the naming format in the Movie_No.txt file. |
| 3 | Confirm that all tiff files conform to the naming format entered in the Movie_No.txt file. |
| 4 | Confirm or change the camera magnification in the MAIN_PROGRAM_MOBILIZED.m and MT_Final_Results.m files. The value must be changed in three locations in the MAIN_PROGRAM_MOBILIZED.m file. |
| 5 | Confirm or change the spur number in the Image_Stack_MT_Skeletonization.m file. |
| 6 | Confirm or change the threshold value in the MT_Skeletonization.m file. |
| 7 | Run the Program from the MAIN_PROGRAM_MOBILIZED.m file. |

The first step in the analysis process is to skeletonize the tiff files. Each tiff file is first converted to a binary black and white image. Any pixel with luminance greater the threshold number is replaced with the value 1 (white) and all other pixels are replaced with the value 0 (black). The result is a single, continuous curve with a width of one pixel. However, some images may still contain small blemishes that branch off of the main curve. If a branch is shorter than the spur number value, then the branch is trimmed. **Figure 2** depicts the output from the skeletonization process for the sample image **Figure 1**. Once the microtubule has been skeletonized the program divides the contour into 23-25 segments of about equal length and calculates: the total number of segments, the exact length of each segment, and the angle of each segment. These properties are used to compute the first 25 modes for each segment

and the variance across the 1,000 frames. The output of the application is a collection of text files and plots detailing: the number of segments each frame is split into, the length of the microtubule in each frame, and the calculated modes.

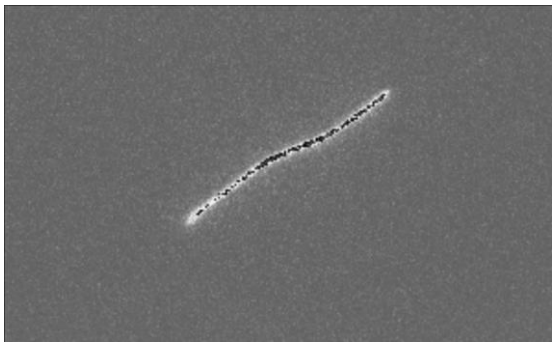

**Figure 1**. Sample image before skeletonization.

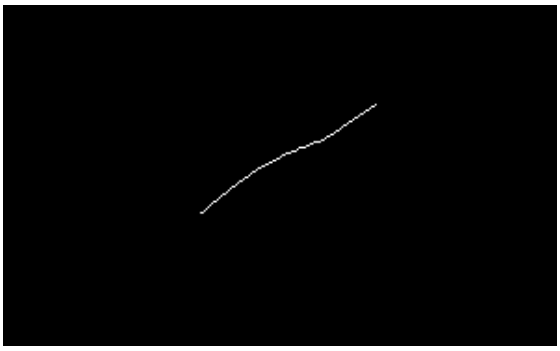

**Figure 2**. Sample image after skeletonization.

While the original MT-Flex code is a vast improvement over performing the analysis by hand, there are some major limitations. Data is passed from one MATLAB function to another by writing the values to a text file and then reading that information back into the other function. The skeletonization step, which generates large matrices of data for each frame, accounts for 58.8% of the 314.07 second total runtime. In addition to increasing the runtime, this step generates 5,992 text and tiff files. Thousands of these text files contain only a single word or number; their only purpose is to pass data between functions. Moreover, **Table 1** lists the multiple locations where values are hardcoded. It is quite easy for a user to overlook one of these locations and generate incorrect data, without being aware of their mistake.

## REQUIREMENTS

The primary objective of modifying the MT-Flex code is to remove the extraneous file reading and writing system in order to decrease the runtime of the application. Any data that is only needed for intermediate steps should be held in memory, rather than written as output. Furthermore, the output that is deemed relevant should be returned in a more useful format. For example, the text files generated by the original application are typically not used in that format by the user. Rather, the contents are manually copied-and-pasted into a spreadsheet for further analysis. As mentioned earlier and depicted in **Table 1**, the multiple hardcoded values located throughout the code are a nuisance for the user to update and are a likely source of error. This issue is remedied rather quickly by storing the values as variables in the main function. Throughout this process other simple opportunities for improvement were uncovered.

## NEW DESIGN

Major changes to the MT-Flex code include: the elimination of file reads, consolidating the functions based on purpose, writing the output in a more useful format, the addition of error-checking functions, and improved console output. While improvements to the user experience are significant, the primary objective of the new design is to improve the runtime of the application. After removing all unnecessary reading of data and restructuring the code, a speedup of 2.019 was achieved. Additionally, after the resource heavy skeletonization step, a speedup of 3.956 for the actual analysis portion of the program was achieved. The performance results are summarized in **Table 2**. All runs were performed on an Apple iMac with a 2.8 GHz Intel Core i7 processor. MATLAB's built-in run-and-time

feature was used to clock the program. Prior to each execution of the code, the entire MATLAB variable workspace was cleared; however, the output files were untouched. This scenario simulates the most common use-case for the program.

**Table 2.** Summary of performance results.

| | **Total Time (s)** | **Time after Skeletonization (s)** |
|---|---|---|
| Original Design ($N$ = 8) | $M$ = 314.072 | $M$ = 128.099 |
| | $SD$ = 5.923 | $SD$ = 3.375 |
| New Design ($N$ = 8) | $M$ = 155.564 | $M$ = 32.514 |
| | $SD$ = 0.764 | $SD$ = 0.931 |
| Speedup | 2.019 | 3.956 |

*N*, number of trials; *M*, mean; *SD*, standard deviation.

The first step in improving the runtime of the code was to remove extraneous file reading and writing. Each function is now defined with the variables that it requires and the data that it returns as parameters. Beyond the improvement in runtime, the new design provides the user with a better understanding of the dataflow throughout the application.

The original fifteen (15) functions were reorganized into eight (8) functions. Five remained as their own function, two were removed entirely, and the rest are combinations of the original functions that are related to each other. The list of files from both versions, fifteen in the old version and eight in the new, are shown in **Table 3**. Each function now accomplishes a specific and clear step in the analysis process. Consequently, there is now minimal data that is required in multiple functions. The steps required to run the new design of the application are summarized in **Table 4**.

**Table 3.** Reorganization of functions.

| **Step** | **New MATLAB Files** | **Original MATLAB Files** |
|---|---|---|
| 0 | mt_main.m<br>→configure.m | MAIN_PROGRAM_MOBILIZED.m |
| 1 | →mt_skeleton.m | →Image_Stack_MT_Skeletonization.m<br>→MT_Skeletonization.m |
| 2 | →mt_skeleton_error_check.m | |
| 3 | →mt_properties.m | → Image_Stack_MT_Properties.m<br>→MT_Properties.m |
| 4 | →mt_properties_plot.m | →MT_Properties_Plot.m |
| 5 | →mt_modes.m | →MT_Modess_1.m<br>→MT_Modess_2.m<br>→MT_Variance_Averages_1.m<br>→MT_Variance_Averages_2.m |
| 6 | →mt_variance_averages_plot.m | →MT_Variance_Averages_Plot.m<br>~~→MT_Variance_Averages_3.m~~ |
| 7 | →mt_estimates_wo_noise.m | →MT_Estimates_woNoise.m |
| 8 | →mt_estimates_w_noise.m | →MT_Estimates_wNoise.m<br>~~→MT_Final_Results.m~~ |

**Table 4.** Summary of how to set up a run with the new design.

| Step | Task |
|---|---|
| 1 | Place tiff files in the *0-frames* folder. |
| 2 | Confirm or change the camera magnification, spur number, and threshold value in the mt_main.m file. |
| 3 | Run the program from the mt_main.m file. |

The functions *Variance_Averages_3* and *Final_Results* have been removed entirely. The role of these functions is to move and rewrite files in the directory in a more user friendly format. However, in the new design the output is always written to the correct location and in the correct format the first-time. This includes writing the results as Comma Separated Value files rather than text files that are then copied into a spreadsheet format later. The reduction of the number of files that are written is summarized in **Table 5**. Additionally, instead of writing thousands of files to the main level of the directory a new folder system has been implemented. Each function that outputs data now has a corresponding folder in the directory. This change does not reduce the runtime of the program, but it does greatly increase the usability of the results.

**Table 5.** Changes in output for step five of the MT-Flex program. Each .csv file contains the same data as the corresponding .txt from the original design, but in a more usable format.

| Purpose of data | Original | New |
|---|---|---|
| Number of segments | 3 .txt files | 1 .csv file |
| Length of microtubule | 1,002 .txt files | 1 .csv file |
| Modes | 2,050 .txt files | 1 .csv file |
| Absolute modes | 1 .txt file | 1 .csv file |
| Average and variance for modes | 25 .txt files | 1 .csv file |

After the major drawbacks of the original application were addressed, some simple, but beneficial areas of improvement were identified. Most notably the original application lacks any error-checking functions. While not necessary for every run of the application, any error-checking process that is easily automated should be left to the application, not the user. First, a configuration function was added in order to ensure that all of the required MATLAB files are present, that the new output files are in place, and that the tiff file folder is non-empty. If any of the output folders are not present, then the configuration function creates them. Additionally, the *skeleton_error_check* function was added to identify any discontinuities in the skeleton images. For any skeleton images that are not continuous, the function returns an error message with the image name and number of discontinuities in the curve. This type of error checking was previously done by scrolling through the images and spotting any discrepancies by visual inspection.

The final simple fix identified was to improve the clarity of the console output. The ambiguous and unorganized console output of the original MT-Flex code is one of the most common complaints about the application. The original output is located in **Appendix A**. The majority of the information included is not relevant to a user who is not familiar with the internal design of the application. The step being run is printed to the console, but the total number of steps is not included. Unless the user is familiar with the program, there is no indication of how much longer the application will run. Additionally, messages from internal MATLAB functions are printed to the console.

These messages are not relevant to a scientist using the application and should be suppressed. The new format is pictured in **Figure 3**. With this format, a user is updated about progress of the program by a brief description of what is accomplished at each step of the analysis process. This console output is useful to both experienced and first-time users.

```
Step 1 of 8: Skeletonize images..................................... Complete
Step 2 of 8: Error check............................................ Complete
Step 3 of 8: Determine skeleton properties.......................... Complete
Step 4 of 8: Plot the first frame................................... Complete
Step 5 of 8: Calculate the first 25 modes and related information... Complete
Step 6 of 8: Plot the modes and amplitudes.......................... Complete
Step 7 of 8: Theoretical expectation for L_p from lower mode........ Complete
Step 8 of 8: Noise contribution and plots to display corrections.... Complete
```

**Figure 3**. Console Output of the new design.

The accuracy of the new design was verified by comparing the output of both versions for a given set of tiff files. Each function was compared individually before the accuracy of the entire application was verified. In addition to comparing the values output to the text files, the multiple plots generated by the application were inspected for any discrepancies.

## CONCLUSIONS

The mechanics of microtubules affect numerous cellular processes. The ability of a microtubule to properly perform these functions is dependent on the stiffness of the microtubule. Persistence length is a mechanical property that quantifies the stiffness of a polymer, such as a microtubule. While the original MT-Flex code is a beneficial tool for studying the persistence length of microtubules, the prolonged runtime and lack of usability create difficulties for the scientists using the application. This paper described the main issues with the original application and how the new design solves them. The new design of the application is 2.019 times faster, performs error-checking itself rather than the user, and returns the results in a more usable format. These changes reduce the amount of time that a scientist must spend on the application before useful data is retrieved. After reorganizing the original workflow, each function now performs a defined step in the image-analysis process. These changes coupled with the improved console output increase the clarity of the application, and reduce the amount of time a scientist much spend learning about the application itself, rather than utilizing it.

## REFERENCES

[1]Megan Bailey, Leslie Conway, Michael W. Gramlich, Taviare L. Hawkins, and Jennifer L. Ross. Modern methods to interrogate microtubule dynamics. *Integrative Biology, 5* (11). 1-6.

[2]Taviare L. Hawkins, Matthew Mirigian, M. Selcuk Yasar, and Jennifer L. Ross. Mechanics of microtubules. *Journal of Biomechanics, 43* (1). 2-7.

[3]Taviare L. Hawkins, Matthew Mirigian, Jingqiang Li, M. Selcuk Yasar, Dan L. Sackett, David Sept, and Jennifer L. Ross. Perturbations in Microtubule Mechanics from Tubulin Preparation. *Cellular and Molecular Bioengineering, 5* (2). 1-8.

# Appendix A

```
running Image_Stack_MT_Skeletonization..
Expecting tiff files with "t" instead of "_" character
If you do not want this change it in Image_Stack _MT Skeletonization.m

running filename: noshear_029t*.tif
Ran Image_Stack_MT_Skeletonization
running Image_Stack_MT_Properties..
Ran Image_Stack_MT_Properties
running MT_Properties_Plot..
moving the skeletonized files...
        Ran MT_Properties_Plot
running MT_Modess_1..
moving the mt_info files...
        Ran MT_Modess_1
running MT_Modess_2..
        Ran MT_Modess_2
        Ran MODES AND DEVIATIONS
running MT_Variances_Averages_1..
        Ran MT_Variances_Averages_1
running MT_Variances_Averages_2..
        Ran MT_Variances_Averages_2
        Ran MAKE AVERAGE AND VARIANCE CALCULATIONS
running MT_Variances_Averages_Plot..
        RAN MT_Variances_Averages_Plot..
running MT_Variances_Averages_3..
        RAN MT_Variances_Averages_3..
running MT_Estimates_woNoise..

Local minimum possible.

lsqcurvefit stopped because the final change in the sum of squares relative to its initial value
is less than the default value of the function tolerance.
<stopping criteria details>

Local minimum possible.
lsqcurvefit stopped because the final change in the sum of squares relative to its initial value
is less than the default value of the function tolerance.
<stopping criteria details>

Local minimum possible.
lsqcurvefit stopped because the final change in the sum of squares relative to its initial value
is less than the default value of the function tolerance.
<stopping criteria details>

        RAN MT_Estimates_woNoise..
                Ran STEP 9
running MT_Estimates_wNoise..

number_of_modes =

4

Local minimum found.

Optimization completed because the size of the gradient is less than the default value of the
function tolerance.

<stopping criteria details>

        RAN MT_Estimates_wNoise..
                Ran STEP 10
running MT_Final_Results..

Set to run on Andor Camera with 60X objective and 2.5 magnification. If not make changes in
MT_Final Results.m
        RAN MT_Final_Results..
                Ran STEP 11
DONE!
```