Manual for the Marathon program

Dmitri Petrov
September 19, 2006

Marathon version: 1.8.5

Options

Data Parameters

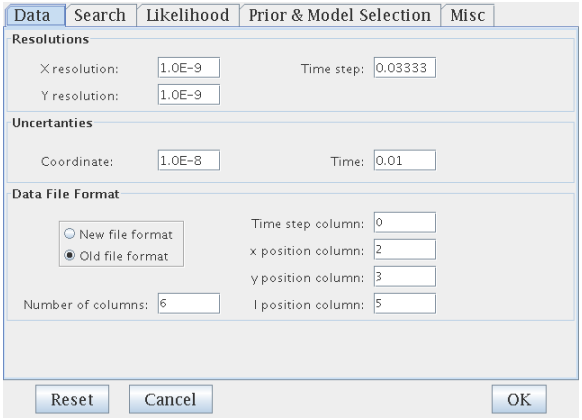

Figure 1: Data file options

Resolutions (Figure 1)

Fields:

- X resolution [m]
- Y resolution [m]
- Time Step [s]

Here we set the units that are used in the data file. Marathon uses SI units for internal data representation, so the units used in the data file must be entered in SI units here. For example, if the coordinates are reported in nanometers, use 1.0×10^{-9} as the x and y resolution. Time step is the time (in seconds) between two successive video frames (or data points). Note that marathon does not use the time column from the data file, instead it uses the product of the index of the data point and the time step to compute the time.

Uncertainties (Figure 1)

Fields:

- Coordinate [m] - uncertainty in determining the X and Y coordinates.
- Time [m] – uncertainty in determining the time.

Uncertainties are the parameters that are used to compute the likelyhood function. Marathon uses the Bayesian approach to find the model (number of segments and their positions) that best fit the data. This approach is based on the following formula

$$P(M|D) = \frac{P(D|M)P(M)}{P(D)} \tag{1}$$

Here $P(M|D)$ is the probability of the model (represented by $M.$) given the data (represented by D). For our case the model is defined by the number of segments and the locations of the ends of the segments (from now on I will refer to them as vertexes). The data is represented by the coordinates (in space and time) of the data points. Clearly our task is to find the model with the highest probability.

The probability of the model is proportional to the likelyhood function $P(D|M)$. This value shows how likely one is to get the data from a given model. To define the likelyhood function one has to assume that the process that generates the data from the model has some uncertainty (or noise) associated with it. In the opposite case the data would have to exactly coincide with the model.

In marathon it is assumed that there are uncertainties in the data in both coordinate and time. These uncertainties do not have to be equal to the errors in determining the position of the cargo because what we are interested in is the position of the motor not the cargo. Thus the uncertainties here mean the uncertainties in determining the position of the motor using the position of the cargo.

Here are some of the details of defining the likelyhood function. At the beginning let's consider the case of just one data point and a single line segment. We assume Gaussian error distribution. This means that the probability of a data point being produced by an infinitely short line segment will look as follows:

$$dp^k \sim \exp\left\{-\frac{1}{2\sigma_x^2}(x - x^k) - \frac{1}{2\sigma_y^2}(y - y^k)\right\} dl. \tag{2}$$

Here x and y are the coordinates of the line segment of length dl, x^k and y^k represent the coordinates of the data point. And σ_x and σ_y represent the uncertainties in determining the coordinates of the data points.

Now we can define a line segment of finite length using the following parameterized equations:

$$\begin{aligned} x &= x_b + \Delta_x t, \\ y &= y_b + \Delta_y t, \qquad t = 0..1. \end{aligned} \tag{3}$$

Using these equations we can compute the probability for a line segment of finite length

$$p^k = A\sqrt{\Delta_x^2 + \Delta_y^2} \int_0^1 \exp\left\{-\frac{1}{2\sigma_x^2}(x_b + \Delta_x t - x^k) - \frac{1}{2\sigma_y^2}(y_b + \Delta_y t - y^k)\right\} dt. \tag{4}$$

The integral in the expression above can be computed analytically (in terms of error functions). After that the extention to the case of multiple data points an multiple segments is trivial.

Data file format (Figure 1)

Fields:

- New file format – if selected, the program assumes that the tracking files are in the new file format. The direction of the microtubule is determined by two numbers in the header.

- Old file format – if selected, the program assumes that the tracking files use the old file format. The direction of the microtubule is given by 0 or 1 in the header of the file. This file format has to have L data.
- Number of columns – total number of columns in the data file.
- Time step column – column where the frame index is recorded.
- x position column – column where the x coordinate of the cargo is recorded
- y position column – column where the y coordinate of the cargo is recorded
- l position column – column where the L (length along the microtubule) coordinate of the cargo is recorded. This is relevant only for files that use the old format.

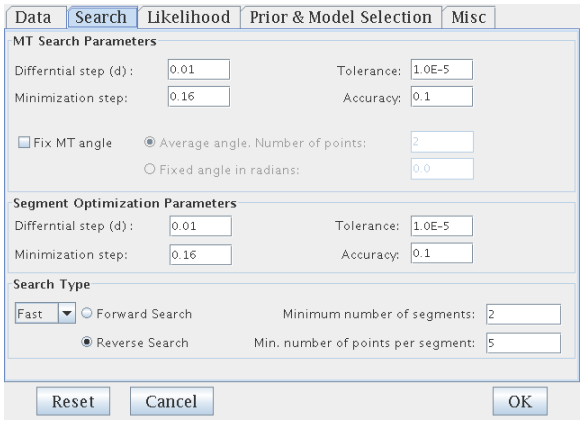

Figure 2: Parameters for the optimization algorithm.

Search Parameters (Figure 2)

MT Search and Segment Optimization Parameters

Important: All values in this tab are given in the units of uncertainty. So, the value that should be entered is the value in SI by the corresponding uncertainty. For example, assume that you want to set some parameter to 1 nm and the uncertainty is 10 nm, in that case you should enter 0.1.

Fields:

- Parameters used in searching for the optimal Microtubule approximation and the optimal positions of the end points of the segments.
  - Differential Step – used to compute derivatives.
  - Minimization Step – initial step for the 1D optimization along the direction of the steepest descent.
  - Tolerance – overall expected accuracy of the optimization.
  - Accuracy – expected accuracy of 1D optimization in the direction of the steepest descent
- Microtubule direction.
  - fix MT angle – if selected the microtubule direction will be fixed using one of two methods explained below.

– Averag angle – the direction is set to the average direction of motion.

– Number of points – the average direction of motion is computed by generating a set of vectors. Each vector points from a data point to its (number of points)th neighbor.

– Fixed angle in radians – the angle between the Microtubule and the X axis will be fixed to the given value.

Here the parameters related to the routine used to search for the optimal positions for the vertexes can be set. The same procedure is used to search for the microtubule and for the optimal positions of the segments, the microtubule is defined as a set of segments consisting of just one segment.

The task of searching for the optimal positions of the vertexes simultaneously is very computationally intensive and would be impossible to perform in reasonable time for a substantially large number of segments. In order to avoid this problem marathon optimizes the positions of the vertexes individually. So, first a vertex is chosen randomly, then its position is optimized. The target precision of this process is controlled by the parameter **Accuracy**. The program decides that it has found the optimal position of the vertex then the fraction of the improvement in the probability for an optimization step becomes smaller then **Accuracy**[1]. After that another vertex is chosen randomly among all the remaining vertexes and its position is optimized. This process continues until positions of all the vertexes are optimized.

The process outlined above constitutes one optimization run. In general, several runs are necessary to find globally optimal positions of all the vertexes. Number of runs is controlled by the parameter **Tolerance**. Just as **Accuracy** this parameter specifies the minimal relative improvement of the probability, but it is related to the improvement produced by the whole optimization run (that consecutively tries to optimize the positions of all vertexes). So, the complete optimization procedure works as follows. First, the probability (negative logarithm of the probability to be exact) is recorded, then an optimization run is performed and positions of all the vertexes are optimized. After that, the new probability is computed. This three steps are repeated until the improvement in probability gets smaller than the **Tolerance** multiplied by the probability. To ensure that the program goes through several optimization runs one should set the **Tolerance** to be much smaller than the **Accuracy**. This will guarantee that the program will find a global maximum (minimum) and won't get stuck in some suboptimal configuration.

The parameters **Differential Step** and **Minimization Step** are used in searching the for the optimal position of a vertex. A variant of the gradient descent algorithm is used for that. The **Differential Step** is the step used in computing derivatives to find a direction of the steepest descent. And the **Minimization Step** sets the initial size of the step that is used to search for the minimum along the direction of the steepest descent. Both of these parameters are given in the units of uncertainties. For example, if the **Differential Step** is 0.01, then the step used to compute the derivative in the x direction will be equal to the uncertainty in determining the x coordinate multiplied by 0.01, the same for the y direction. (Here x and y directions can be time and length).

The **Minimization Step** can not be smaller than the **Differential Step**. In general, it should be some integer that can be expressed as a power of two multiplied by the **Differential Step**. The result of the program should not be very sensitive to these two parameters as long as they are reasonable.

Search Type (Figure 2)

Fields:

- Selector: Fast/Slow – determines what algorithm is used for adding and removing segments. The slow algorithm is more accurate, that is when using this algorithm the probability that after removing or adding a vertex the system will be in a configuration close to optimal is

[1]Note that instead of maximizing the probability $P(M|D)$ marathon minimizes the following value: $-ln(P(M|D))$. The results are equivalent, but the parameters **Accuracy** and **Tolerance** set the minimum relative improvement in the logarithm of the probability, not the probability itself.

higher. This means that in some cases selecting the Slow algorithm will actually improve overall performance (and accuracy!).

- Forward Search – if selected the program searches for the optimal number of segments by fitting the track by a few segments and then increasing the number of segments. See Note below.
- Reverse Search – opposite of the Forward Search.
- Minimum Number of segments – defines the starting point for the Forward search and the end point for the Reverse Search.
- Min. Number of points per segment – defines the end point for the Forward search and the starting point for the Reverse Search.

The search type determines how the program tries to find the optimal number of segments. The **Forward Search** starts with low number of segments and tries to find the optimal configuration by successively splitting existing segments in half. After a segment is split in half the positions of all vertexes are optimized and the new value of the posterior probability $P(M|D)$ is computed. After this procedure is completed the optimal number of segments is determined using one of the two methods: the **calibration** procedure and the **compensation for fluctuations**. They are both described in the **Prior & Model Selection** section of this manual.

The **Reverse Search** starts with large number of segments and proceeds to try to eliminate unnecessary segments. Just as in the **Forward Search** after a segment is eliminated the positions of the remaining segments are optimized and the new probability is computed.

The parameter **Minimum number of segments** sets the initial number of segments for the **Forward Search** and the final number of segments for the **Reverse Search**. Conversely, the parameter **Min. Number of points per segment** defines the maximum number of segments that can be used to fit a track.

NOTE: The **Reverse Search** procedure works much better than the **Forward Search**. So, use the **Reverse Search** unless you have a good reason not to. I will not be responsible, if you choose to use **Forward Search** and receive bogus results.

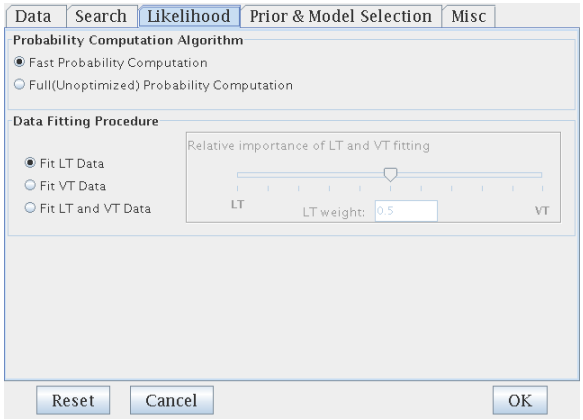

Figure 3: Parameters for computing the likelihood function.

Likelihood (Figure 3)

Probability Computation Algorithm

Fields:

- Fast Probability Computation – use this
- Slow (Unoptimized) Probability Computation – for debugging only

This parameter does not affect the results produced by the program. If **Full Probability Computation** is chosen the probabilities of all the segments are recomputed every time the probability of the model is computed. This procedure is simple, but slow and a lot of unnecessary computations are performed. If **Fast Probability Computation** is selected, only the probabilities of the segments that have moved since the last probability computation are recomputed. This procedure is much more efficient and fast, and produces the same result. However, it is much more complex and there is a possibility of bugs, that is why the **Full Probability Computation** is still there.

Data Fitting Procedure

Fields:

- Fit LT Data
- Fit VT Data
- Fit LT and VT Data

Here we select the data type that is fitted to the constant velocity segments. The most tested and trusted approach is to fit the Length vs Time data (**Fit LT Data**). Alternatively one can choose to fit the instantaneous velocities (**Fit VT Data**). In this case the instantaneous velocities between successive data points are computed first and then fitted by a set of segments of constant velocity. Finally, one can try to fit both length and velocity.

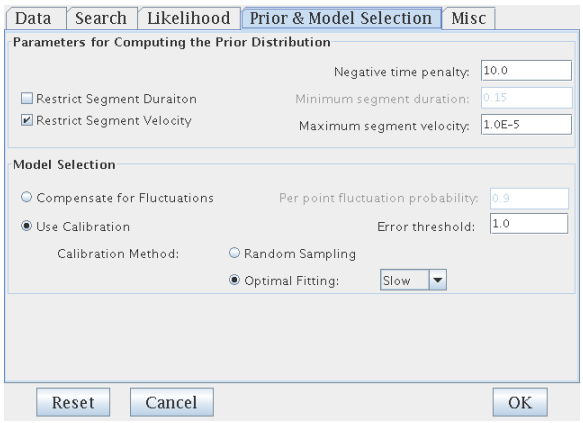

Figure 4: Parameters for computing the prior probability.

Prior & Model Selection (Figure 4)

Parameters for Computing the Prior Distribution

Fields:

- Negative time penalty – Penalty for having segments with negative duration
- Restrict segment duration
- Minimum segment duration – if 'Restrict segment duration' is selected you can specify the minimum segment duration here.

- Restrict segment velocity
- Maximum segment velocity – if 'Restrict segment velocity' is selected you can specify the maximum absolute value of velocity here.

This section contains parameters that affect the computation of the prior distribution. In equation 1 the prior distribution is labeled by $P(M)$. It represents our knowledge about the model that is independent of the data. For example, we know that motors can not go faster than certain velocity, so that information can be included in the prior distribution.

The prior distribution is used by Marathon to limit the duration and velocity of segments. First, we ensure that there are no segments with negative duration or, alternatively, that there are no segments shorter than the duration specified by the user. The **Negative time penalty** parameter is related to the penalty the model has to suffer if any of the segments have negative duration or are shorter than the user defined duration. If the **Restrict Segment Duration** option is selected then the user can choose the minimum duration of the segment to be equal to the number specified as the **Minimum segment duration**.

Second, the maximum velocity of a segment can be limited by selecting the **Restrict Segment Velocity** option. If this option is selected, the prior probability of any segment that has velocity higher than **Maximum segment velocity** will be reduced by the amount proportional to the **Negative time penalty** and the difference between the segment velocity and the maximum velocity threshold defined in **Maximum segment velocity**.

Model Selection

Fields:

- Compensate for Fluctuations – Do not use Calibration. Number of segments is determined using an 'ad hoc' approach.
- Per point fluctuation probability – Wee below.
- Use calibration – use a calibration curve to determine the optimal number of segments.
- Error threshold – see below
- Calibration method: Random sampling – Construct the calibration curve by using random data points as the vertices of the segmented line. No optimization is performed.
- Calibration method: Optimal Fitting – Construct the calibration curve by fitting the data by segmented lines consisting of different numbers of segments.

In the current context the term **Model Selection** refers to the method of determining the optimal number of segments. There are two alternative procedures implemented in Marathon that can be used to achieve this goal.

The first approach, that can be selected by choosing **Compensate for fluctuations** option, uses the prior distribution to give advantage to longer segments. The reasons is that longer segments have higher probability of being interrupted due to noise, so we have to try to compensate for that by increasing the prior probability of longer segments. If we assume that the chance that a single data point was affected by a significant fluctuation is q, then the probability that a segment of duration corresponding to n data points was not affected by a fluctuation is $1 - q^n$. The parameter q is called in marathon **Per point fluctuation probability**.

The second approach, that is selected by choosing the **Use Calibration** option, can be used when tracking data of a cargo, which does not exhibit any directed motion caused by molecular motors, is available. This data can be used to calibrate the program. After that the most probable number of segments in a track with motor motion is determined by comparing the probability of

the model for a given number of segments with the calibration value. If the probability is lower than the calibration value, more segments are needed to correctly represent the data. So the most probable number of segments is equal to the minim number of segments that produce a model with the probability comparable to the calibration value.

The calibration curve can be determined by either fitting the calibration data by models with different numbers of segments **Optimal Fitting** of choosing random data points as vertexes **Random Sampling**. My current testing suggests that **Optimal Fitting** is a much better choice, using **Random Sampling** is not advised. When **Optimal Fitting** is used it uses the Reverse or Forward approach depending on what is selected in Search → Search Type.

The parameter **Error threshold** specifies how far the probability value has to be from the calibration value to be "comparable". The value **Error threshold** is given in the units of the standard deviation of the calibration values. When the calibration curve is constructed, the standard deviation is also estimated.

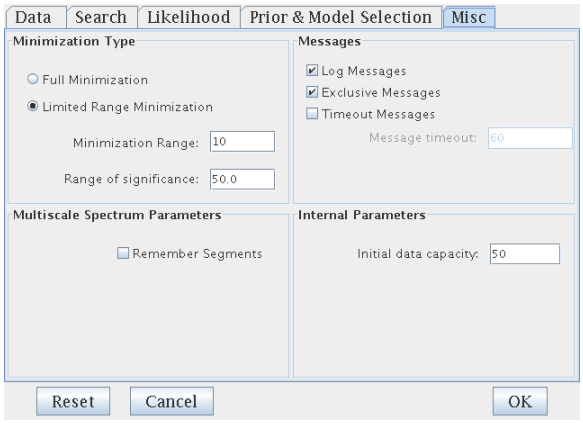

Figure 5: Parameters that do not fit into any 'special' category.

Miscellaneous Parameters (Figure 5)

Minimization Type

Fields:

- Full Minimization – positions of all vertices are optimized at every step.
- Limited Range Minimization – only positions of segments that are 'close' to a segment that was removed or added are optimized.
- Minimization Range – range (in vertices) that determines a set of vertices that will be optimized.
- Range of Significance – see below

The parameters presented here are used in speeding up the program at the expense of some possible accuracy loss. However, I tested the program with and without these optimizations enabled and could not detect any significant difference in the results, provided the default parameters were used.

If the **Full Minimization** option is selected, then at every optimization run when the program tries to find optimal position of the vertexes it will try to optimize the position of every vertex. However, if the **Limited Range Minimization** is selected, then after the program is initialized it will optimize the positions of all the segments. After that whenever a segment is added or removed,

the program will only try to optimize the position of that vertex and **Minimization Range** vertexes before and after it.

The **Range of Significance** is given in the units of the time uncertainty. The idea behind this optimization is that if a segment starts much later or ends much earlier than the time at which a data point was recorded, then the probability of that data point to be caused by this segment is zero and there is no need to compute it.

Messages

This panel is used to control the behavior of the error messages produced by Marathon.

If **Log Messages** checkbox is set, all messages are logged in a file Marathon.log. This file is created in the directory from which the program was started. This is not the same directory as the directory where the segment files are saved.

If **Exclusive Messages** checkbox is set, only one message can be displayed at a time. If new message appears, the old message is removed automatically.

If **Timeout Messages** checkbox is set, messages automatically disappear in **Message timeout** seconds.

Multiscale Spectrum Parameters

Multiscale spectrum is a curve that shows the dependence of the fitting quality on the number of segments. Objects representing the multiscale spectra are extensively used by the program when the 'calibration curve approach' is used. Usually when a Multiscale spectrum is computed the configuration of a segmented line (vertex positions) is discarded and only the number of segments and the quality is recorded. The option **Remember Segments** tells the program to save the vertex positions in the multiscale spectrum. This significantly increases the memory footprint of the program, but allows one to efficiently process the tracking series without having to recompute the multiscale spectra. For example, a different calibration curve can be applied to the tracks.

Internal Parameters

The parameter **Initial Data Capacity** the storage space allocated by default for storing the locations of the vertexes. This can grow as needed, so this parameter does not have much effect and should not be bothered with, unless you run on a system with very limited memory, then you can make it smaller.

Usefull Hints

The font size in the main window and in the terminal window can be changed by pressing the '-' and '=' buttons. ('-' makes font smaller, '=' makes it larger).