Cohort Explorer v2018.05

Hodgkin Lab
Walter and Eliza Hall Institute of Medical Research

HoChan Cheon

February 15, 2019

Contents

1 Overview

Cohort Explorer is an analysis tool that provides a quick exploration of the cell cloning dynamics through cohort method. It automatically estimates and extracts cell kinematic characteristics (such as division rate, time to enter first division etc) from the data, and plot them. You can study basic idea of cohort method from the poster, and extensively from publications [2, 5, 7]. This document assumes that you are familiar with fundamental principles and concepts of cohort method.

We implemented common form of Levenberg-Marquardt (LM) optimisation algorithm in order to find best fit for the data. Following list shows the mathematical models and associated implicit assumptions used in the analysis. Together with models and algorithm, the program runs series of fitting procedures and returns key estimations in full details including 95% confidence interval.

1. *Discrete Gaussian function - normality of distribution*
   Key estimation: **total cohort number** (amplitude of distribution, $\hat{A}$), **mean division number** ($\hat{\mu}_{\text{div}}$), and **standard deviation** ($\hat{\sigma}_{\text{div}}$).

2. *Piecewise linear function with unknown breakpoint - linear correlation & existence of plateau*
   Key estimation: **time to enter subsequent division**, **time to enter first division**, **division destiny**, and **time to enter division destiny**.

3. *Cumulative distribution function - normality of distribution*
   Key estimation: **mean time to enter first division**.

Alternatively, the program also carries simple arithmetic cohort operations in parallel, which endures much less assumptions than that of curve fitting method. It could be useful for datasets that are relatively far off from the models or lack of data points, and provide a secondary option to pick up insight on proliferation kinetics. Both results are conveniently collected in one plot panel so that you can easily compare two independent methods. In depth details of both methods are discussed in section (3).

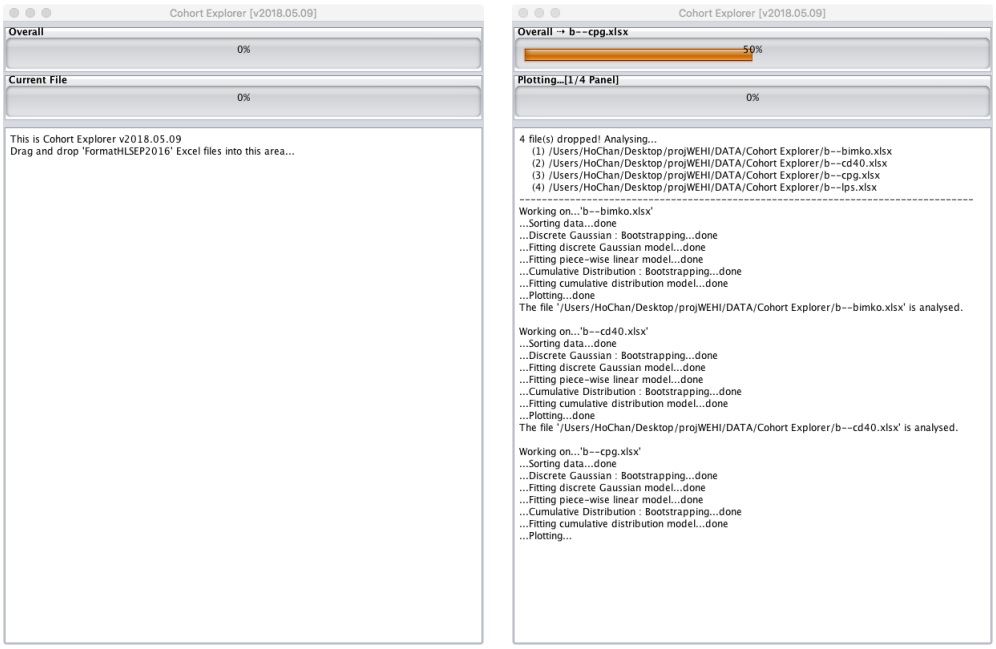
1.1 Quick Start

This application was developed in Java environment as such you need a Java Virtual Machine (JVM) installed in the system. Most of modern computers have it pre-installed so there is a good chance that your system already has it but ensure that latest Java is installed on the computer. Installation can be done by following steps:

1. Download the software from this link.
2. Extract the zip file.
3. Open [Cohort Explorer v2018.05.jar] file.
4. A window should pop up as shown in figure(1a).
5. Try drag and drop example excel files included in the package.

Multiple files can be dropped at once as shown in figure(1b) but note that the program does not handle input files concurrently. The overall progress bar indicates percentage of completed files (out of all the input files) and the right arrow shows the current working file. The second progress bar displays the intermediate computation progress.

List of files and each computation steps are printed to the main screen in order to provide feedbacks and keep track of operations to the user. Should the program encounters any errors (e.g. unexpected excel format), it will terminate the analysis and reported back to user with clear error messages. Most common



(a) Main window (b) Analysing state

Figure 1: Cohort Explorer Application

errors are raised due to data format so it is recommended to read Section 1.3 thoroughly and interrogate with provided example data before getting into the real one.

You will notice that the program automatically generates *settings.txt* file if you run it for the first time. This is to set number of bootstrap iterations for calculating 95% confidence intervals on every parameter estimations. More iteration number would contribute better parameter range, but it does not generally improve overall accuracy of the estimation. Keep in mind that higher iterations could result in expensive computation time, so adjust it for a good compromise between adequate report of confidence interval and speed.

1.2 Outputs

There are three main output routines when analysis is done; (i) A secondary interactive window (figure 2) to collect and display all the plots, (ii) one excel file that contain data points used for plotting as well as estimated parameters, (iii) individual plots in SVG and collected plots in PNG formats.

All output files can be found on newly created directory named after input data file. Meta information such as time stamps, version of the program, and settings are printed in *info.txt* for future reference. Note that output directory will not be overwritten should you run the program with same input file.

Figure numbers represent tab number in the interactive panel, and subfigures are labelled alphabetically (left to right, top to bottom in the panel) as a suffix to the figure number. Exported plots are named after their corresponding figure number and alphabets for consistency. First tab of the panel shows evolution of the raw cell number counts for overview of the dynamics (square points: mean data $\pm$ S.E.M). If number of beads are present in your data it will calculate adjusted total cell number.

We then extract snapshots of the cell evolution, and plot them in the second tab in terms of cohort number as a function of generation. Each plot contains fitted discretised Gaussian curve with its estimated parameters displayed in respective legend (round bracket: 95% confidence interval from bootstrap). Since generation 0 often contains noisy information (e.g. unstimulated/dead cells), we excluded this from best-fit

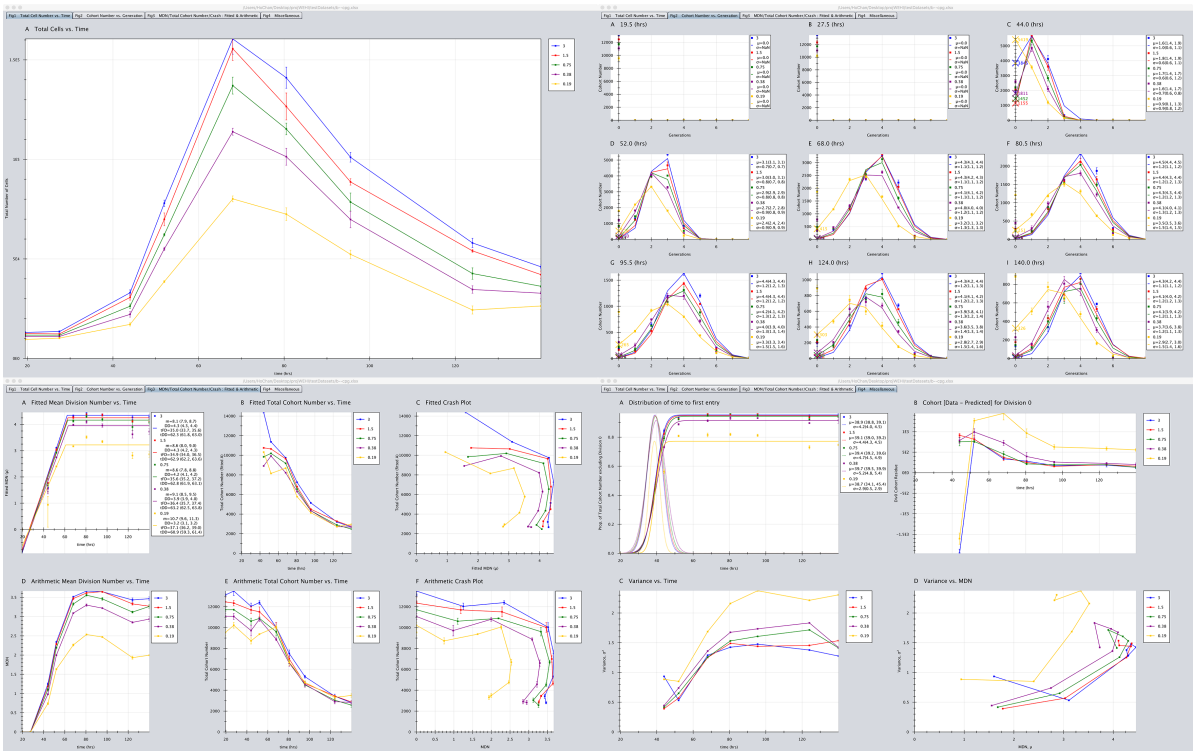


Figure 2: Interactive secondary window from example excel file “b–cpg.xlsx”

and over plotted it with extrapolated generation 0 (marked as “X”) along with the actual data.

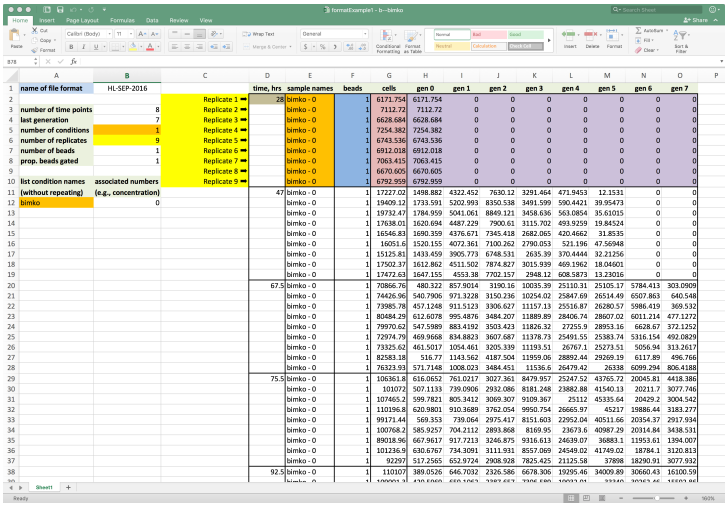
These results are then passed on to third tab to create summary plots of parameter evolution, and perform secondary piecewise fitting in order to draw more in depth cell kinematic parameters (tab 3 plot A). Notice that this tab is mainly for comparing two different mathematical approaches introduced in Overview section. The first row is a collection of fitted results from tab 2, and the second row is results from independent arithmetic cohort method.

Lastly, we fit cumulative distribution function (CDF) to ratio of sum of cohort number (excluding generation 0) to total cohort number (tab 4 plot A). This would effectively give us accumulative number of cells that are entering first generation. Therefore, it is possible to infer a distribution of average time to enter first division. Solid line is fitted CDF curve and dashed line is its complementary Gaussian distribution. Tab 4 plot B, C, and D are residue of extrapolated generation 0 and data, fitted variance as function of time, and variance against fitted mean division number respectively. These are information carried from tab 2, discrete Gaussian fitting results.

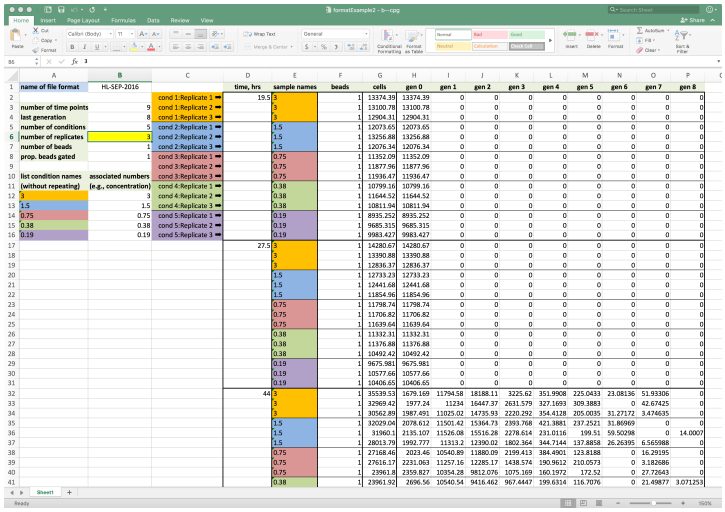
1.3 Excel data format

We have adopted a custom excel format (HL-SEP-2016) in order to match output style of FlowJo, a cell proliferation assay analysis software, so that you can simply copy and paste into a new or existing excel file. In this way, we can minimise amount of manual operations to prepare data, and potentially human errors. Column A comprises of experimental setup information whereas from column D to P are the main part of the cell count data. It is important to flag the excel file exactly as “HL-SEP-2016” in column B, row 1 because the program will look for this particular cell to confirm that input file is indeed in a correct format. This was implemented for future updates and backward compatibility should we introduce a new data format. An example data is shown in figure (3).

Note the order of coloured blocks for differentiating various conditions. Each time point must have row



(a) One condition experiment with 9 replicates per condition



(b) Multiple conditions with 3 replicates per condition

Figure 3: FormatHLSEP2016 Excel File

number equal to number of conditions times number of replicates in an order that you specified in list condition names. If you have a missing replicate for particular condition, then leave the row blank to flag it as an empty replicate. The program will automatically adjust calculation (e.g. average cell number) during analysis.

1.4 Flow of Computation

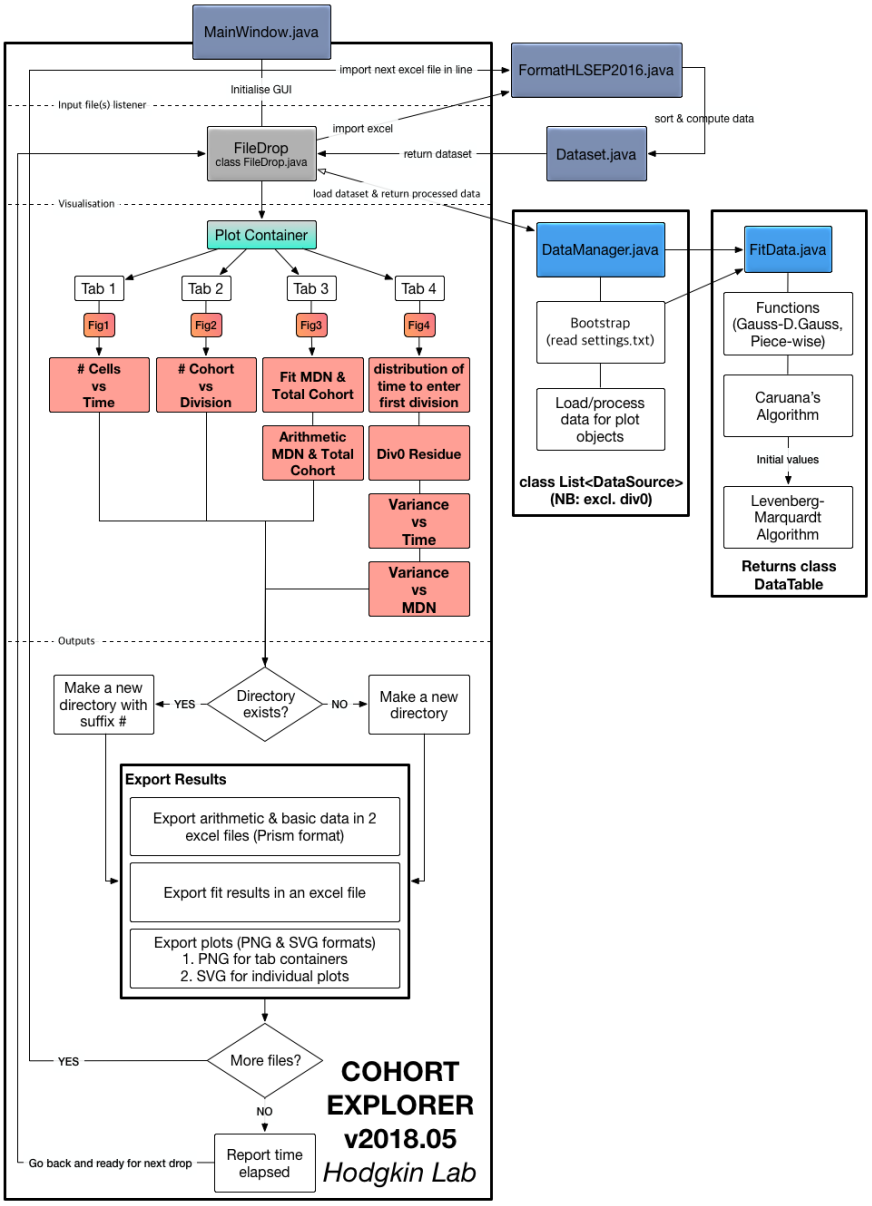


Figure 4: Cohort Explorer - Flow Chart

2 Preliminary Mathematics of Cohort Analysis

Let $n(k,t)$ be number of cells at generation k and at time t, then the cohort number is defined as

$$c(k,t) = \frac{n(k,t)}{2^k}, \tag{1}$$

for a cell divides into two cells as it subsequently enters next generation. This allows us to construct a distribution of cohorts per time as a function of generation, cf. figure(5, 6). Such distribution can be quantitatively analysed by two seemingly different methods, *fitting* and *arithmetic*, but ultimately shares same goals; computing mean division number (MDN), time to first division(tFD), division destiny(DD), and potentially many other biologically significant quantities of proliferation dynamics.

3 Fitting Method

3.1 Least-squares Problem

Given a set of data, $d_i = (x_i, y_i)$, and a model function, $f(x;\hat{\mathbf{p}})$, we can obtain parameters that minimise sum of square residues of the form,

$$\underset{\hat{\mathbf{p}}}{\operatorname{argmin}} F(\hat{\mathbf{p}}) = \underset{\hat{\mathbf{p}}}{\operatorname{argmin}} ||\mathbf{r}(x_i;\hat{\mathbf{p}})||^2 = \sum_{i=1}^{N} [y_i - f(x_i;\hat{\mathbf{p}})]^2 \tag{2}$$

where N is the number of data, $\hat{\mathbf{p}}$ is a parameter vector that its elements are minimisers of $r(x_i,\hat{\mathbf{p}})$. This is famous least-squares problem. Depending on linearity of a model function, different numerical technique is required. Simple linear functions involve solving a linear system of equations (e.g. $\mathbf{A}x = \mathbf{B}$) in which it is often solved by deploying linear algebra packages (or even analytically solved), whereas non-linear cases generally demand iterative procedure.

For our regression cohort analysis, we chose following list of non-linear functions for best fits,

1. Discrete Gaussian function
2. Piece-wise linear function with unknown breakpoints
3. Cumulative distribution function

All of which are non-linear with respect to corresponding parameters, thus, we implemented iterative numerical recipe called Levenberg-Marquardt(LM) algorithm [1, 3, 6] with an aid of Caruana's method [4, 9] as it is well accepted non-linear least-squares problem solver. It is frequently performed with constrains (with prior knowledge on experiments or mathematical anomalies) that bind parameter range but we did not implement any restriction to the algorithm.

As powerful as the algorithm can go, it also suffers from one major limitation. LM algorithm requires a set of initial guesses at the beginning of iteration, and convergence is known to be susceptible to the choice of starting values, especially for noisy data (relative to the model) and complex functions. We handle this problem by implementing a systematic procedure to generate educated guesses in order to minimise underfitting and adapt to input data as much as possible.

3.2 Caruana's Algorithm

The advantage of using Caruana's algorithm is that it does not require initial guesses. The theory begins with linearising unnormalised Gaussian function, $f(x;\hat{A},\hat{\mu},\hat{\sigma}) \equiv y = \hat{A}e^{-(x-\hat{\mu})^2/2\hat{\sigma}^2}$ by taking natural logarithm on both sides and expand to yield,

$$\ln(y) = \ln(\hat{A}) - \frac{\hat{\mu}^2}{2\hat{\sigma}^2} + \frac{\hat{\mu}}{\hat{\sigma}^2}x - \frac{1}{2\hat{\sigma}^2}x^2 \tag{3}$$

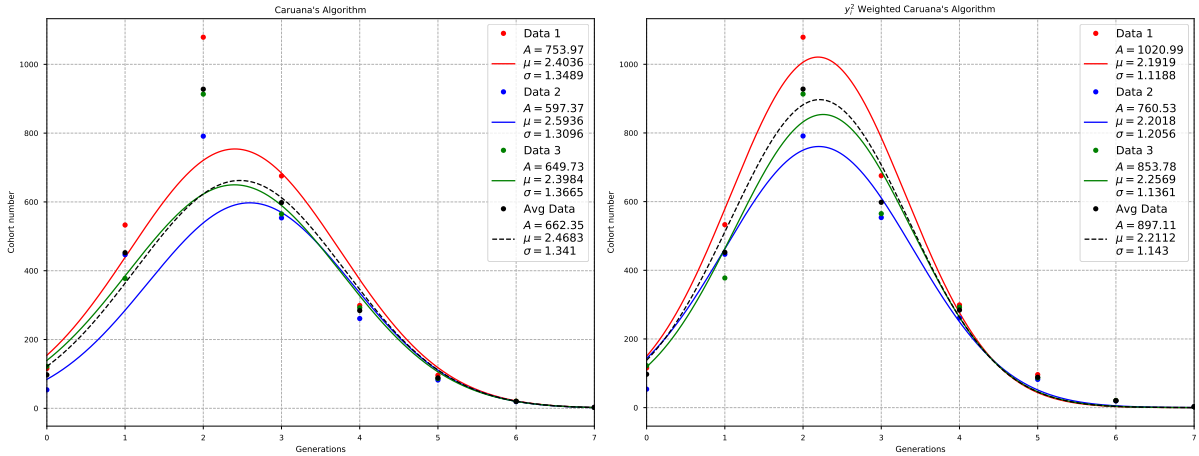


Figure 5: **Example results of Caruana's algorithm.**

Let $\hat{a} = \ln(\hat{A}) - \hat{\mu}^2/2\hat{\sigma}^2$, $\hat{b} = \hat{\mu}/\hat{\sigma}^2$, and $\hat{c} = -1/2\hat{\sigma}^2$, then equation is reduced to a second degree polynomial function,

$$\ln(y) = \hat{a} + \hat{b}x + \hat{c}x^2 \tag{4}$$

So our residue function to minimise is,

$$r(x_i; \hat{a}, \hat{b}, \hat{c}) = \ln(y_i) - (\hat{a} + \hat{b}x_i + \hat{c}x_i^2) \tag{5}$$

Taking derivatives of r^2 with respect to $\hat{a}$, $\hat{b}$, and $\hat{c}$, and letting resultant expressions to zero produce linear system of equations,

$$\begin{bmatrix} N & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{bmatrix} \begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix} = \begin{bmatrix} \sum \ln(y_i) \\ \sum x_i \ln(y_i) \\ \sum x_i^2 \ln(y_i) \end{bmatrix} \tag{6}$$

where $\sum$ runs through N number of observed data. We can weight this equation by y_i^2 to yield,

$$\begin{bmatrix} \sum y_i^2 & \sum x_i y_i^2 & \sum x_i^2 y_i^2 \\ \sum x_i y_i^2 & \sum x_i^2 y_i^2 & \sum x_i^3 y_i^2 \\ \sum x_i^2 y_i^2 & \sum x_i^3 y_i^2 & \sum x_i^4 y_i^2 \end{bmatrix} \begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix} = \begin{bmatrix} \sum y_i^2 \ln(y_i) \\ \sum x_i y_i^2 \ln(y_i) \\ \sum x_i^2 y_i^2 \ln(y_i) \end{bmatrix} \tag{7}$$

After solving equation (7), parameters $(\hat{a}, \hat{b}, \hat{c})$ are converted back to original Gaussian parameters,

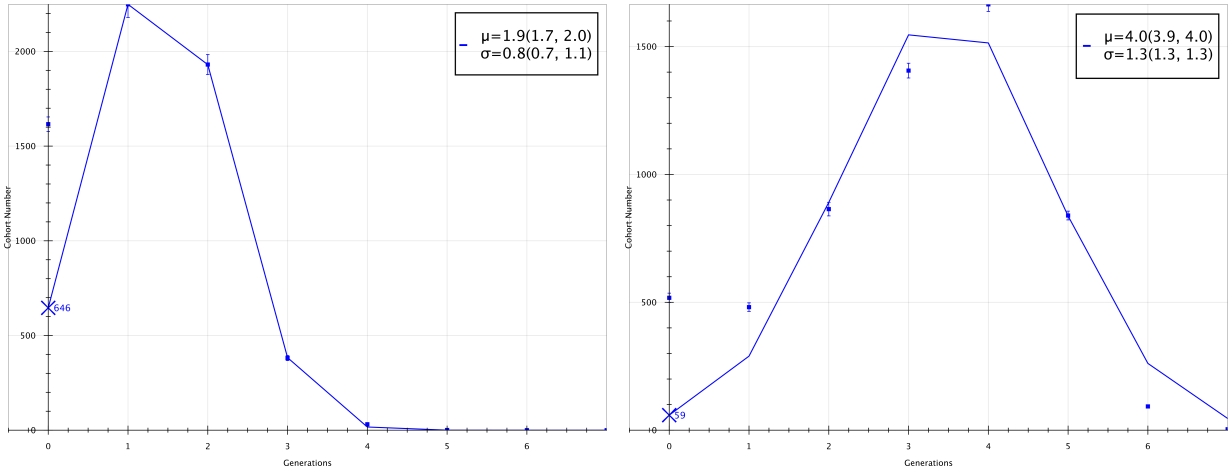
$$\hat{A} = \exp\left[\hat{a} - \frac{\hat{b}^2}{4\hat{c}}\right], \qquad \hat{\mu} = -\frac{\hat{b}}{2\hat{c}}, \qquad \hat{\sigma} = \sqrt{-\frac{1}{2\hat{c}}} \tag{8}$$

Both unweighted and weighted Caruana fit results are shown in Figure (5). It is not ideal algorithm for datasets that are heavily deviated from Gaussian distribution as it propagates errors logarithmically should the errors exist. However, this method is computationally inexpensive, and it can provide reasonable initial starting points for more sophisticated fitting algorithm.

3.3 Discrete Gaussian Fitting

Recall unnormalised normal distribution,

$$\mathcal{N}(k; \hat{A}', \hat{\mu}_{\text{div}}, \hat{\sigma}_{\text{div}}) = \hat{A}' \exp\left[-\frac{(k - \hat{\mu}_{\text{div}})^2}{2\hat{\sigma}_{\text{div}}^2}\right] \tag{9}$$



(a) Early time point: $\hat{\mu}_{\text{div}} = 1.9^{+0.1}_{-0.2}$ $\hat{\sigma}_{\text{div}} = 0.8^{+0.3}_{-0.1}$ (b) Next time point: $\hat{\mu}_{\text{div}} = 4.0^{+0.0}_{-0.1}$ $\hat{\sigma}_{\text{div}} = 1.3^{+0.0}_{-0.0}$

Figure 6: **Fitting equation (11) to observed data collected at two different time points.** Solid Line - best fit, Data points = mean ± S.E.M. "X" label is extrapolated generation 0 cohort number.

where k is generation, and $(\hat{A}', \hat{\mu}_{\text{div}}, \hat{\sigma}_{\text{div}})$ are the parameters to be estimated. It is reasonable model by itself for estimating distribution of cohorts, however, there is an experimental limitation we need to take into account. During the process of sorting data from flow cytometry, it is commonly practiced to bin cell counts by examining carboxyfluorescein succinimidyl ester (CFSE) profile. The profile generally shapes multimodal distribution, figure (7), in which an operator semi-automatically gates cell counts per generation around the peaks. Inevitably, it is often hard to distinguish the exact boundaries of generations as cells do not discretely divide. Consider a cell in a mid way process to enter next generation (e.g. at generation 0.6). Because cells are continously distributed around the discrete generation number, essentially we are collapsing the datapoint as following:

$$\begin{aligned}
\text{gen0} \leq \text{data point} < \text{gen0.5} &\rightarrow \text{gen0} \\
\text{gen0.5} \leq \text{data point} < \text{gen1.5} &\rightarrow \text{gen1} \\
\text{gen1.5} \leq \text{data point} < \text{gen2.5} &\rightarrow \text{gen2} \\
&\vdots
\end{aligned}$$

Evidently this would induce a selection bias toward the dataset, and propagates through remaining cohort analysis. This motivates us to apply mathmetical correction via discretising Gaussian model by integrating through consecutive generations.

$$f(k; \hat{A}', \hat{\mu}_{\text{div}}, \hat{\sigma}_{\text{div}}) = \int_{k}^{k+1} \mathcal{N}(k'; \hat{A}', \hat{\mu}_{\text{div}}, \hat{\sigma}_{\text{div}}) dk' \tag{10}$$

which is equivalent to

$$f(k; \hat{A}, \hat{\mu}_{\text{div}}, \hat{\sigma}_{\text{div}}) = \left(\hat{A}\sqrt{2\pi\hat{\sigma}^2_{\text{div}}}\right) \frac{1}{2} \left[\text{erf}\left(\frac{k+1-\hat{\mu}_{\text{div}}}{\hat{\sigma}_{\text{div}}\sqrt{2}}\right) - \text{erf}\left(\frac{k-\hat{\mu}_{\text{div}}}{\hat{\sigma}_{\text{div}}\sqrt{2}}\right)\right] \tag{11}$$

where $\left(\hat{A}\sqrt{2\pi\hat{\sigma}^2_{\text{div}}}\right)$ is the total cohort number of particular dataset, and $\hat{\mu}_{\text{div}}$ is MDN that describes average generation of the cells.

Cell tracking technique can only detect cells up to certain generation (usually generation 7) before fluorescence dye that traces cell proliferation gets too diluted to be detected. This implies that "last generation"

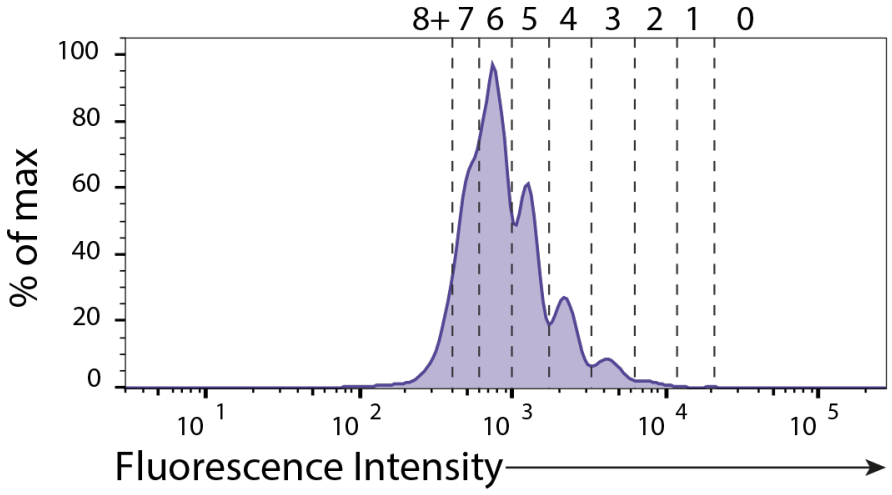


Figure 7: **CFSE profile using CellTrace Violet dye.** Cell division profile and gating of cell counts per generation shown as dash-lines. **CITE Tempany, Zhou, Bryant and Hodgkin**

is not a definitive termination of cell division, but rather experimental limitation on detecting (perhaps) existence of further generations. Hence, we implemented an additional correction to the data as following,

$$c(k = \text{last}, t) = \frac{\hat{n}(k,t)}{2^k} = \frac{1}{2^k}\sum_{i=k}^{\infty} 2^i \hat{c}(i) \tag{12}$$

where $\hat{n}(k,t)$ is estimated total cell number beyond last generation, and $\hat{c}(i)$ is an extrapolated cohort number from the model, $f(k; \hat{A}, \hat{\mu}_{\text{div}}, \hat{\sigma}_{\text{div}})$. It is obvious that the sum ideally needs to extend to the infinity but for computational purpose, we truncated it to 30 generation for a good approximation.

We repeatedly perform above corrections and fitting for all time points given in data, and the set of estimated $\hat{\mu}_{\text{div}}$ is then passed to construct a MDN time series plot, which then ultimately fitted to piece-wise linear function.

3.4 Piece-wise Linear Fitting

At first glance, one can expect that simple linear regression algorithm could be applied by splitting two regions and attach the estimated functions together. This can only be done if the break point is known prior to the fitting. However, it is generally an unknown quantity and is part of our main question to be answered.

We modified equation taken from Marsh et al. [8], and run LM algorithm to explore parameter space of

$$g(t; \hat{a}, \hat{b}, \hat{bp}) = \hat{a} + \hat{b}\cdot t \cdot D_1 + \hat{b}\cdot \hat{bp} \cdot D_2 \begin{cases} (D_1, D_2) = (1, 0), & \text{if } t \le bp \\ (D_1, D_2) = (0, 1), & \text{if } t > bp \end{cases} \tag{13}$$

Note that equation (13) is defined to be continuos at the break point but not necessarily differentiable. Once we have obtained $(\hat{a}, \hat{b}, \hat{bp})$ values, we can estimate time to enter subsequent division(m), time to first division(tFD), division destiny(DD), and time to division destiny(tDD):

$$\text{m} = \frac{1}{\hat{b}}, \qquad \text{tFD} = \frac{1-\hat{a}}{\hat{b}}, \qquad \text{DD} = \hat{a} + \hat{b}\cdot \hat{bp}, \qquad \text{tDD} = \hat{bp} \tag{14}$$

Since it is a secondary fitting procedure passed down from previous section, there could be hidden errors potentially propagated/amplified in the process. Consider following case: if majority of cells are not entering first division stage, then our cohort distribution is hardly look like a Gaussian distribution. This generally

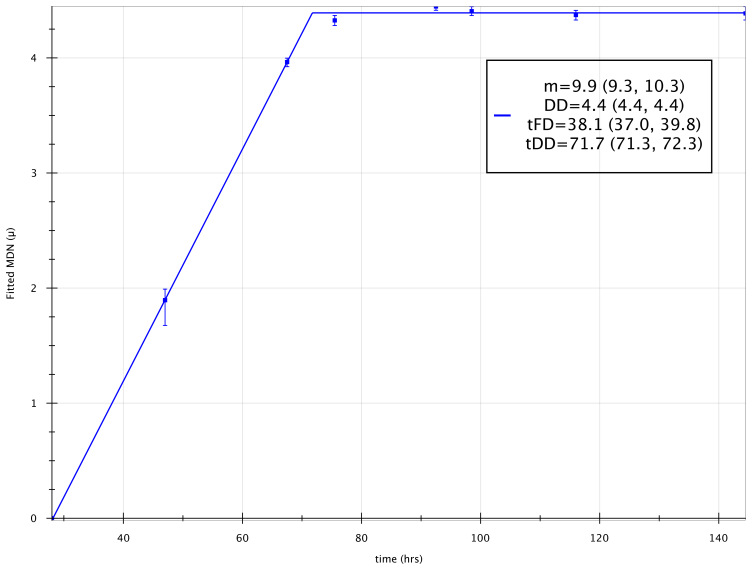


Figure 8: **Piece-wise linear fitting with estimated set of $\hat{\mu}_{\text{div}}$.** Error bars are 95% confidence interval computed from bootstrapping. m is time taken to enter subsequent division (k $\rightarrow$ k+1), DD is generation that cell reaches its terminal division, tFD is time when the cell enters first division, and tDD is the break point that describes time when the cell enters division destiny.

results in large 95% confidence interval range, therefore, affects piece-wise fitting. In order to minimise this, we adjust LM algorithm to incoroperate errors by weighting each points with standard deivation computed by bootstrapping method.

4 Arithmetic Method

In this section, we show alternative approach to compute previously derived quantities without any assumptions about the distribution. It is a simple arithmetic calculation in order to obtain statistical moments but not necessarily for further parameter estimation. Total cohort number is summation of equation (1) for all generation,

$$S(t) = \sum_{k=0}^{\infty} c(k,t) \tag{15}$$

This is equivalent to amplitude of the fitted discrete Gaussian function. Then we calculate average generation by summing weighted cohort number with respective generation number and divide the sum by total cohort number,

$$h(t) = \sum_{k=0}^{\infty} \frac{kc(k,t)}{S(t)} \tag{16}$$

Using equation (16), we can obtain similar plot to figure (8). Notice that arithemtic and fitting methods produce different results even though they share same biological property. There are three critical reasons,

1. Arithmetic method does not make correction (discretisation) to the dataset. This is a major factor that generally shifts MDN values.

2. Dataset exhibits non-Gaussian distribution. In this particular case, estimated MDNs are unreliable, and it is shown by broad confidence interval (i.e. error bars).

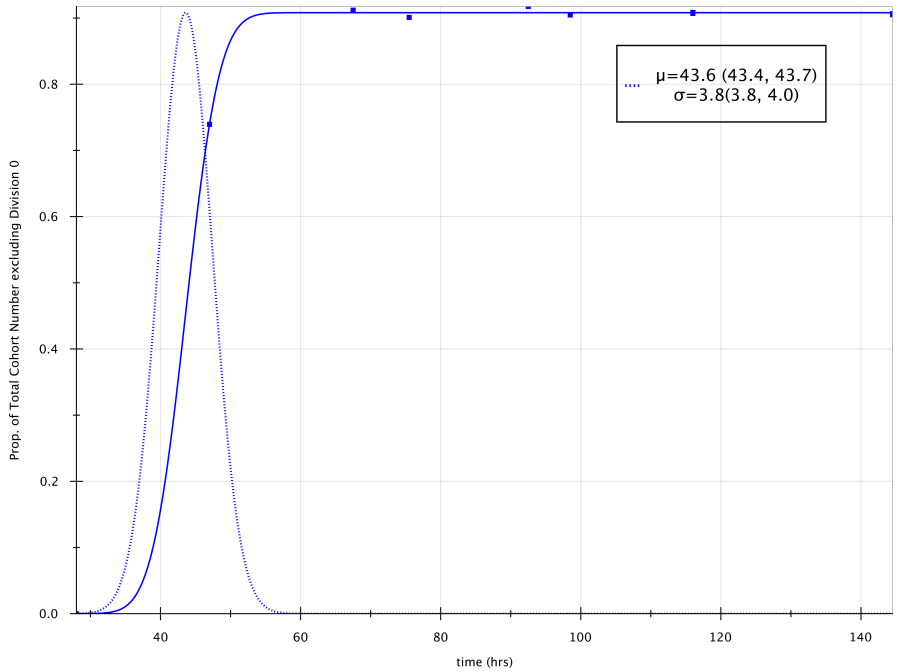


Figure 9: **An estimated distribution of time to enter first division.** Solid line is best-fit curve to CDF. Dotted line is complementary (continuous) Gaussian distribution. Bootstrap method is utilised to acquire 95% confidence intervals on each estimated parameters.

3. Fitting method can explore negative MDN region. It is a feature of unconstrained best-fit algorithm, which we take advantage of in order to investigate regime otherwise impossible to extrapolated by arithemtic method. This is debatable on how to interpret negative MDN estimations.

It becomes a non-trivial problem to generalise and make realiable fitting algorithm for all possible datasets. As such, we provide outcomes from least assumed method side by side with fitting method so that you can choose a method that works best for your dataset.

4.1 Cumulative Distribution Fitting

This is only case where we fit arithmetically computed quantities to estimate a distribution of time to enter first division. Basic idea is that we examine change in ratio of total cohort number excluding division 0 to equation (15). As cells start to divide and enters first division, the ratio approaches a certain value as division 0 cells are cumulatively move into subsequent division.

$$M(t) = \frac{S_{\mathrm{ex0}}(t)}{S(t)} = \frac{\sum_{k=1} c(k,t)}{\sum_{k=0} c(k,t)} \tag{17}$$

if we ignore measurement errors that result in differnt total cell counts per time points, theoretically, it is a bounded function $M(t) \in [0,1]$. We then fit this equation to cumulative distribution function,

$$\Phi(t; A, \mu, \sigma) = \frac{A}{2}\left[1 + \mathrm{erf}\left(\frac{t-\mu}{\sigma\sqrt{2}}\right)\right] \tag{18}$$

It is again an alternative routine that comparable to parameter tFD in equation (14).

References

[1] Croeze A., L. Pittman & W. Reynolds (2012) *Solving nonlinear least-squares problems with the Gauss-Newton and Levenberg-Marquardt methods*. https://www.math.lsu.edu/system/files/MunozGroup1%20-%20Paper.pdf

[2] Gett A.V. & Hodgkin P.D. (2000) *A cellular calculus for signal integration by T cells*. Nature Immunology.

[3] Griva I., Nash S.G. & Sofer A. (2009) *Linear and nonlinear optimization*. Philadelphia, Pennsylvania: siam

[4] Guo H. (2011) *A simple algorithm for fitting a Gaussian function [DSP Tips and Tricks]*. IEEE Signal Processing Magazine - IEEE SIGNAL PROCESS MAG. 28. 134-137. 10.1109/MSP.2011.941846.

[5] Hawkins E.D. et al. (2013) *Quantal and graded stimulation of B lymphocytes as alternative strategies for regulating adaptive immune responses*. Nature Communications.

[6] Madsen K., Nielsen H.B. & Tingleff O. (2004) *Methods for non-linear least squares problems*. IMM Lecture Note retrieved from http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3215/pdf/imm3215.pdf

[7] Marchingo J.M. et al. (2014) *Antigen affinity, costimulation, and cytokine inputs sum linearly to amplify T cell expansion*. Science.

[8] Marsh L., Maudgel M. & Raman J. (1990) *Alternative methods of estimating piecewise linear and higher order regression models using SAS software*. Proceedings of SAS Users Group International 15: 523–527.

[9] Pastuchova E. & Zakopcan M. (2015) *Comparison of Algorithms For Fitting a Gaussian Function Used in Testing Smart Sensors*. Journal of Electrical Engineering. 66. 178-181. 10.2478/jee-2015-0029.