Quantal analysis

Background

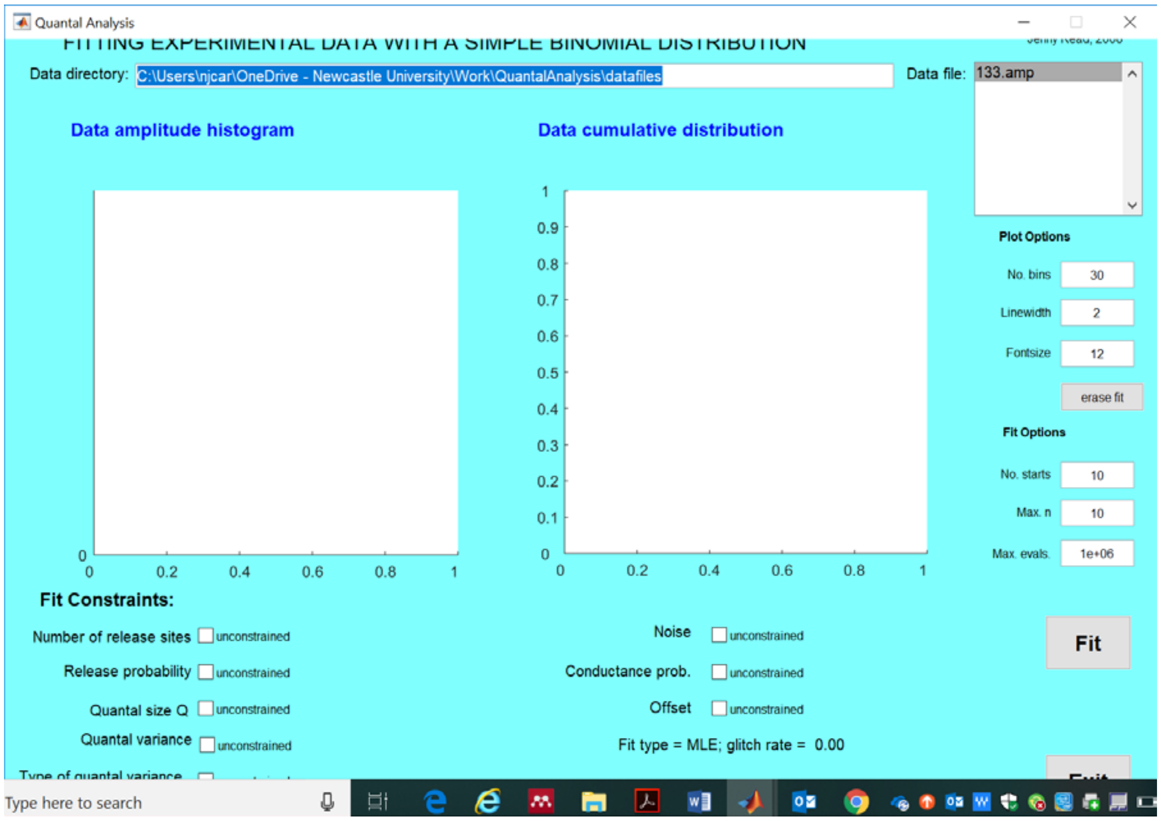
From Hardingham et al 2006:
"Histograms of amplitude frequency distributions of excitatory postsynaptic potential (EPSPs) from stable periods of data often (~50% of recordings) contained regularly spaced peaks, indicative of a quantal release of neurotransmitter at the synapses. It has been shown that neocortical synapses relating to individual connections appear to operate with similar release probabilities, which are target derived (Koester and Johnston, 2005) and so can be approximated with a simple binomial model. Therefore, the working hypothesis was that the EPSP amplitudes were drawn from a simple binomial distribution characterized by the number of release sites (*n*), release probability (*p*), and quantal size (*q*) (Larkman et al., 1997)."

This program attempts to fit n, p and q, along with some other parameters.

Getting started

FitData6 is the "main" file -- Type FitData6 at the command window to start things off.

In the Matlab command window, go to the directory where the files are and type "FitData6". The following window should open:



Data directory: C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles

Data file: 133.amp

No. bins 30
Linewidth 2
Fontsize 12
No. starts 10
Max. n 10
Max. evals 1e+06

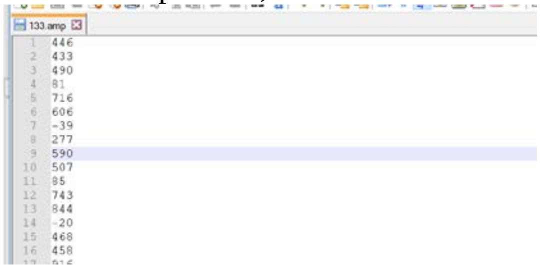
Number of release sites ☐ unconstrained
Release probability ☐ unconstrained
Quantal size Q ☐ unconstrained
Quantal variance ☐ unconstrained
Noise ☐ unconstrained
Conductance prob. ☐ unconstrained
Offset ☐ unconstrained

Fit type = MLE; glitch rate = 0.00

Experimental data

The data should be in text files with one EPSP value per row. The "default directory" field of the GUI tells the program where to look for these. Results files get written there too, so you can look up what fits were tested and compare the results of different methods.

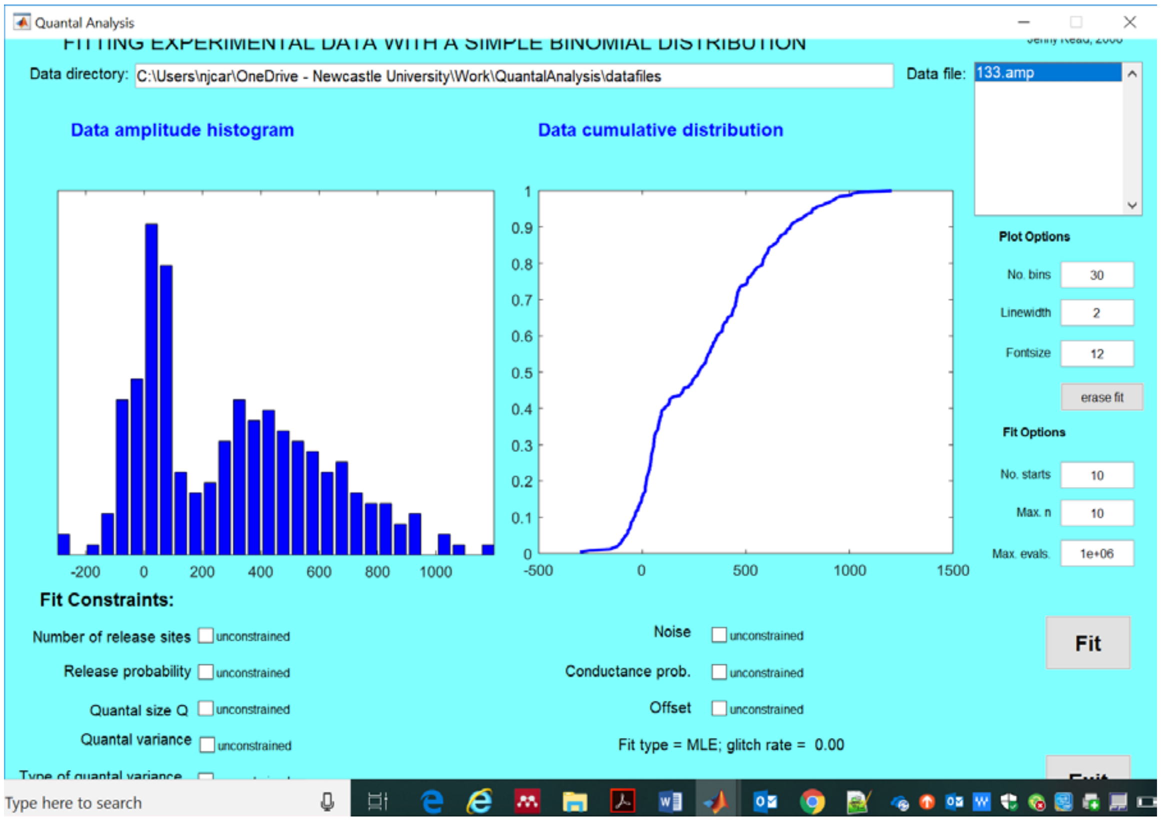
Type the location of your datafiles into the “Data directory” window.
This will bring up a list of available .amp files. This should be a text file containing a list of EPSP amplitudes, one on each row of the file, e.g.



133.amp

```
446
433
490
81
716
606
-39
277
590
507
85
743
844
-20
468
458
```

Click on the file you want to select its data. The data will then be displayed:



FITTING EXPERIMENTAL DATA WITH A SIMPLE BINOMIAL DISTRIBUTION

Data directory: C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles

Data file: 133.amp

Plot Options

No. bins 30

Linewidth 2

Fontsize 12

Fit Options

No. starts 10

Max. n 10

Max. evals. 1e+06

Fit Constraints:

Number of release sites ☐ unconstrained

Release probability ☐ unconstrained

Quantal size Q ☐ unconstrained

Quantal variance ☐ unconstrained

Noise ☐ unconstrained

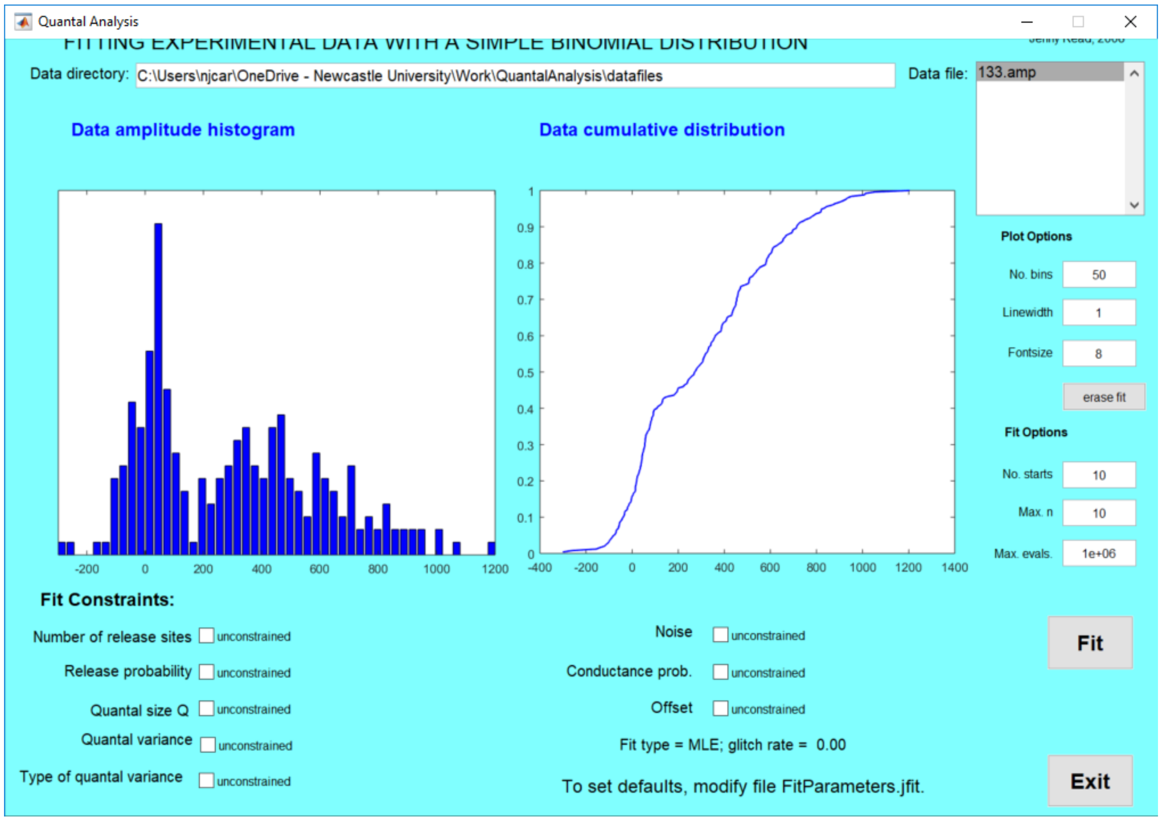
Conductance prob. ☐ unconstrained

Offset ☐ unconstrained

Fit type = MLE; glitch rate = 0.00

Plot options

In “Plot Options” on the right, you can change how the data is plotted:



FITTING EXPERIMENTAL DATA WITH A SIMPLE BINOMIAL DISTRIBUTION

Data directory: C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles

Data file: 133.amp

No. bins: 50
Linewidth: 1
Fontsize: 8

No. starts: 10
Max. n: 10
Max. evals: 1e+06

Fit Constraints:

Number of release sites ☐ unconstrained
Release probability ☐ unconstrained
Quantal size Q ☐ unconstrained
Quantal variance ☐ unconstrained
Type of quantal variance ☐ unconstrained
Noise ☐ unconstrained
Conductance prob. ☐ unconstrained
Offset ☐ unconstrained

Fit type = MLE; glitch rate = 0.00

To set defaults, modify file FitParameters.jfit.

Note that this has *no* effect on the fits. The number of bins is purely for display purposes. Fits are done to the raw data, not to binned data. This is why I show the cumulative distribution– it doesn't require any choices about bins.

Model

The probability density function for the EPSP size v is:

$$f(v) = p_{stim} \sum_{k=1}^{n} \frac{n!}{m!(n-m)!} p^m (1-p)^{n-m} \frac{1}{\sigma_k \sqrt{2\pi}} \exp\left(-\frac{(v - v_0 - mq)^2}{2\sigma_k^2}\right) + (1 - p_{stim}) \frac{1}{\sigma_{noise}\sqrt{2\pi}} \exp\left(-\frac{(v - v_0)^2}{2\sigma_{noise}^2}\right)$$

v = amplitude of EPSP
v_0 = offset, assumed to be added to all EPSPs. This allows for the fact that the mean amplitude of failures may differ slightly from zero, because of extracellular field effects (Stricker et al., 1996).
σ_{noise} = SD of experimental noise, assumed to be Gaussian
n = number of release sites.
q = quantal size
p = release probability at each release site
p_{stim} = probability that stimulation results in an action potential which reaches the release sites

σ_q = quantal variance; =SD on first peak in absence of noise.
σ_m = variance affecting the *m*th peak; *m* ranges from 0 to n.

For Type I quantal variance, $\sigma_m^2 = \sigma_{noise}^2 + m\sigma_q^2$;
for "flat" quantal variance, $\sigma_m = \sigma_{noise}$ for m=0; $\sigma_m^2 = \sigma_{noise}^2 + \sigma_q^2$ for m>0.

The idea behind the above equation is:
You have a recording between a pair of connected neurons, neuron A and neuron B. When neuron A fires an action potential, in general you record an EPSP in neuron B. This EPSP represents the release of *m* vesicles of neurotransmitter, each contributing an average amount *q* to the EPSP (a quantum). Two complications are

(i) there may be a offset v_0 such that the EPSP amplitude is measured as being non-zero even when no vesicles are released. v_0 is fixed for a given recording but varies between recordings.
(ii) quanta are not exactly the same on every occasion. With Type 1 quantal variance, we assume that each quantum is drawn from a Gaussian with mean q and SD σ_q. In this case, if m quanta are released, their total effect is drawn from a Gaussian with mean mq and SD $\sigma_q\sqrt{m}$. With "flat" quantal variance, we assume that the SD is independent of the number of quanta released (above 0), so that the total effect of m quanta is drawn from a Gaussian with mean mq and SD σ_q.

We further assume that the total effect is then subject to Gaussian noise.

The number of quanta released depends on two factors.
First, an action potential in neuron A may somehow fail to reach the synapse. We define p_{stim} to be the conductance probability, i.e. the probability that the action potential reaches the synapse. If it does not reach the synapse, 0 quanta are released and the EPSP amplitude v is drawn from a Gaussian with mean v_0 and noise σ_{noise}.
If the action potential does reach the synapse, then each of the n release sites has probability p of releasing one quantum (whose amplitude is drawn from a Gaussian distribution as described above).

This model results in the probability distribution given above.

Glitch rate

At one point I modified the code to allow for a glitch rate *g*. Any EPSP generated by the above quantal model has a probability *g* of being replaced with a random number between g_{min} and g_{max}. In my code, g_{min} is set to be 1 less than the lowest EPSP in the data-set, and g_{max} to be 1 more than the highest. So the effect of g depends to some extent on the data-set. I don't regard this as a big problem, since the whole point is that the best fit should be largely independent of g, except possibly if g=0.
The final PDF is then

$$f_{final}(v) = (1-g)f(v) + \theta(v-v_0)\, g/(g_{max}-g_{min}),$$

where $\theta(v)$ is 0 for $v<g_{min}$ and $v>g_{max}$, and 1 elsewhere.

If glitchrate is set to 0, it will have no effect. However, if you come across an example where you think one outlier is throwing things off, try setting glitchrate to 0.01 or 0.05, and see if that improves things. NB if glitchrate is non-zero, then the program draws two fits over the data histogram. The one in red is what was actually fitted, ie the model including the non-zero glitchrate. The dotted pink line is what the PDF would look like if all the parameters were the same except glitchrate was 0.

However I subsequently hard-wired the glitch rate to 0, which is where it is currently.

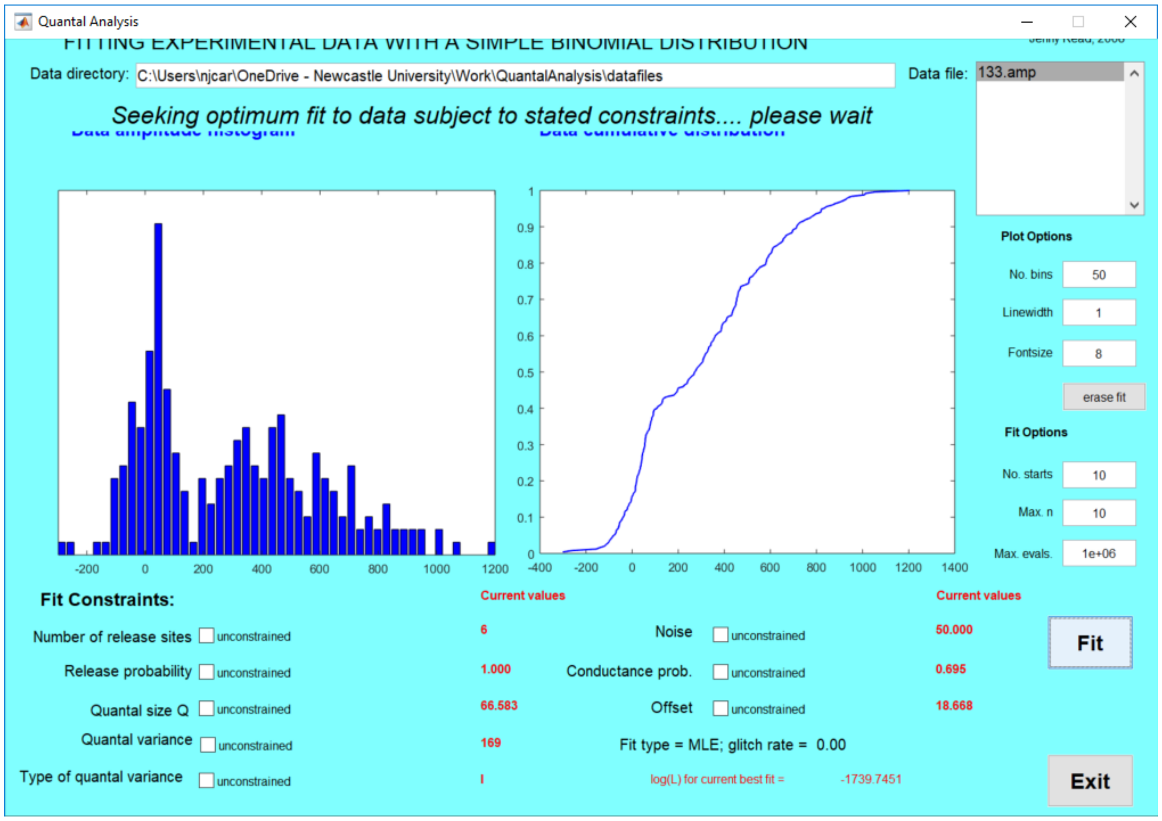
Fitting

When you hit the Fit button, the program will try and fit model parameters to the data. It uses the method of maximum likelihood, i.e. it tries to find the parameters which make the data most likely given the model.
There are potentially 8 free parameters:
n = number of release sites, an integer between 1 and n_{max}, set in the GUI.
p = release probability, a real number between 0 and 1
q = quantal size, a positive real number
σ_q = “quantal variance” (actually quantal SD), a positive real number
Quantal variance type, a binary variable which describes how σ_q is used (Type 1 or flat)
σ_{noise} = experimental noise, a positive real number
p_{stim} = conductance probability, a real number between 0 and 1
v_0 = offset, a real number

The fitting is done with FMINSEARCH; see Matlab's help on that. NB it minimises not maximises, so everything's upside-down -- I actually minimise -sum(log(p)) instead of maximising sum(log(p)) etc.

Fitting usually takes a while. The program goes through all the allowed values of n, from 1 to n_{max}, and maximizes the other fit parameters. It will then report the value of n that gave the maximum likelihood.

This is what the GUI looks like while it’s fitting:


Quantal Analysis
FITTING EXPERIMENTAL DATA WITH A SIMPLE BINOMIAL DISTRIBUTION
Data amplitude histogram
Data cumulative distribution
Plot Options
Fit Options
erase fit
Fit
Exit

Data directory: C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles

Data file: 133.amp

Seeking optimum fit to data subject to stated constraints.... please wait

No. bins: 50
Linewidth: 1
Fontsize: 8

No. starts: 10
Max. n: 10
Max. evals.: 1e+06

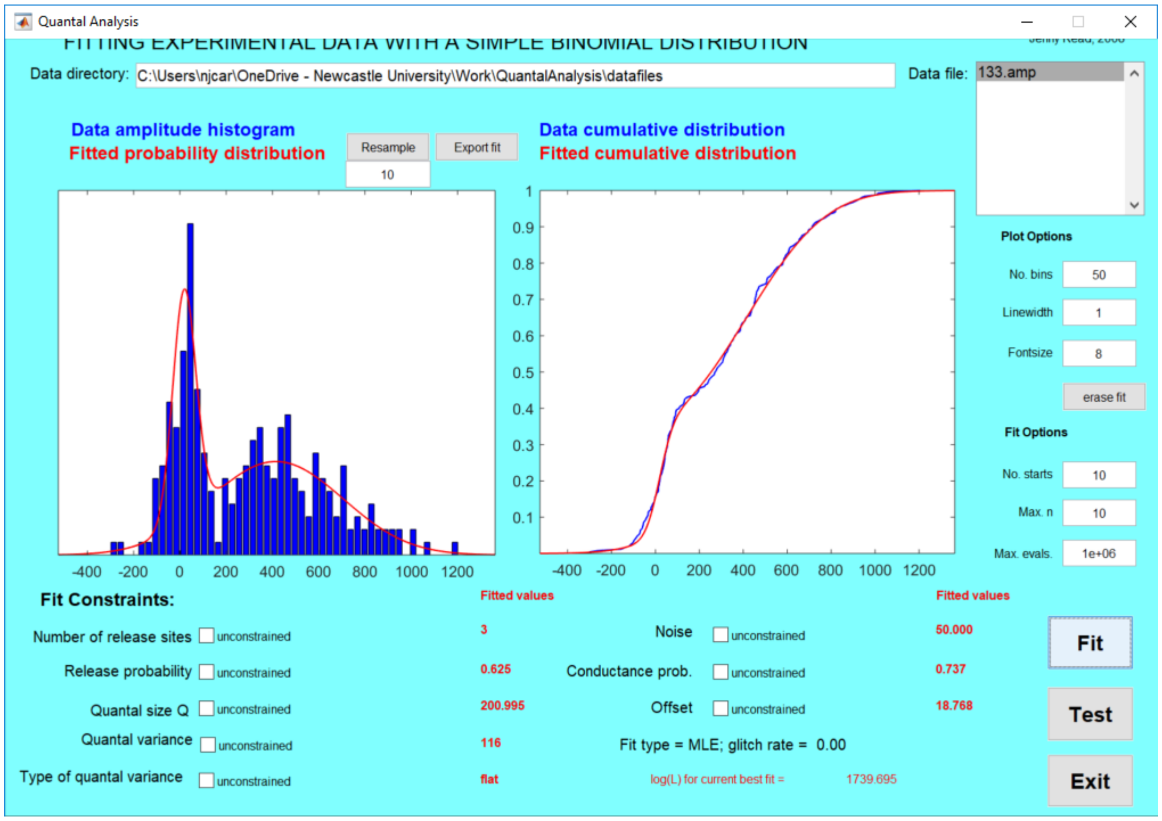
Fit Constraints:

		Current values			Current values
Number of release sites	☐ unconstrained	6	Noise	☐ unconstrained	50.000
Release probability	☐ unconstrained	1.000	Conductance prob.	☐ unconstrained	0.695
Quantal size Q	☐ unconstrained	66.583	Offset	☐ unconstrained	18.668
Quantal variance	☐ unconstrained	169			
Type of quantal variance	☐ unconstrained	I			

Fit type = MLE; glitch rate = 0.00

log(L) for current best fit = -1739.7451

When it’s finished, it will draw the model in red for you over the data:



Data directory: C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles

Data file: 133.amp

Resample: 10

No. bins: 50
Linewidth: 1
Fontsize: 8

No. starts: 10
Max. n: 10
Max. evals: 1e+06

		Fitted values			Fitted values
Number of release sites	unconstrained	3	Noise	unconstrained	50.000
Release probability	unconstrained	0.625	Conductance prob.	unconstrained	0.737
Quantal size Q	unconstrained	200.995	Offset	unconstrained	18.768
Quantal variance	unconstrained	116			
Type of quantal variance	unconstrained	flat			

Fit type = MLE; glitch rate = 0.00

log(L) for current best fit = 1739.695

Here, in a fully unconstrained fit, the model decided there were probably 3 release sites, with a quantal size of around 200uV and release probability 0.625. The conductance probability was 0.7, meaning that on around a third of trials, the action potential didn't make it to the synapse. That explains the big peak at zero.

Things you can change about fitting

You can change the maximum number of release sites, "Max n" in the GUI. There is generally no point fitting more than 10, as you don't find connections with 11 clear peaks. Making "Max n" bigger will slow down run time and won't necessarily improve your results. Keep it as low as your physiology suggests is sensible.

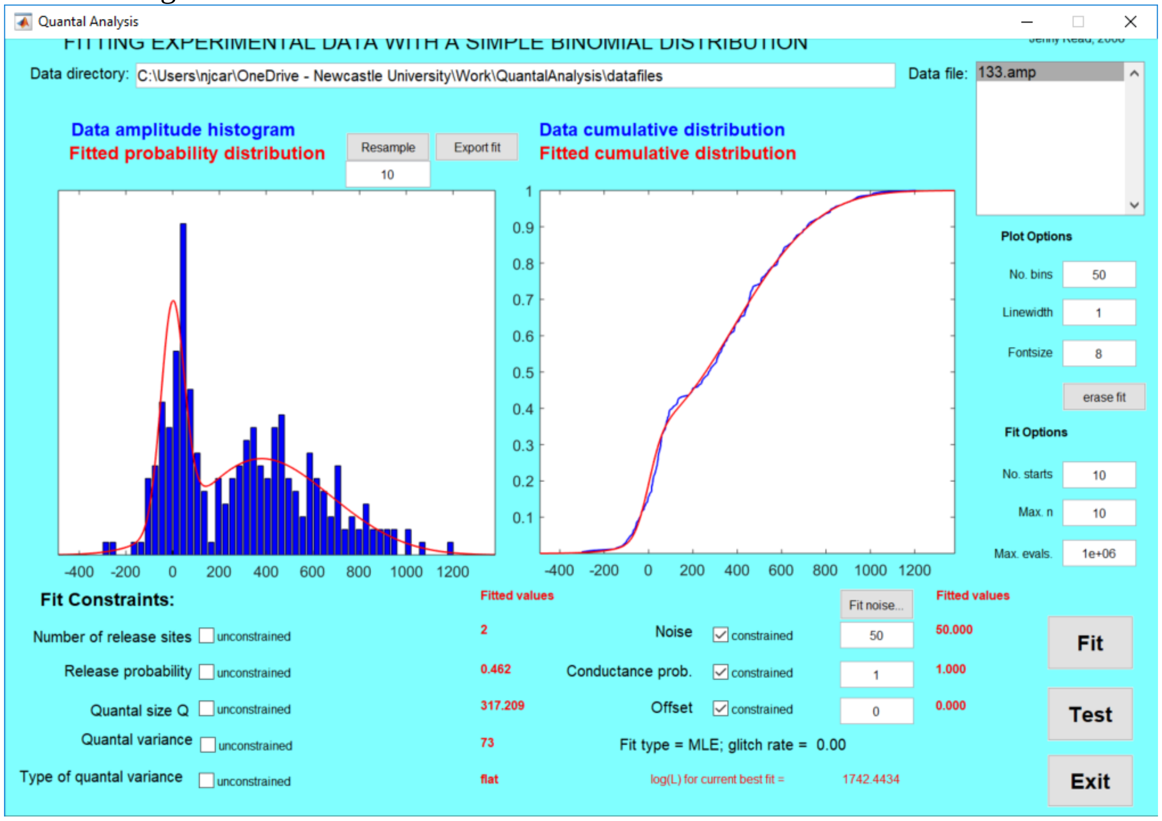
You can change the number of randomly-chosen starting points for the fitting. This helps to avoid the algorithm getting stuck at local optima. This is "No. starts" in the GUI. Making it bigger makes your results more reliable but the run time slower.

You can change the maximum number of function evaluations in fminsearch – see Matlab's documentation. That's "Max. evals" in the GUI. Again, making it bigger makes your results more reliable but the run time slower.

You can also constrain some of the parameters. This is generally a good idea if possible, as it will make run times faster and fits more reliable. For example, you may be able to estimate the experimental noise independently, in which case feed that into the fit.

Note that if you only have two peaks in your histogram, probably there is only one release site. In that case **it makes absolutely no sense to fit the conductance probability.** There is probability (1-p_{stim})+p_{stim}(1-p) that 0 quanta are released, and probability p_{stim}p that 1 quantum is released. You have no way of distinguishing between p_{stim} and p. So if your data looks as if you have only one release site, you should definitely set p_{stim}=1.

In the example below, say I estimated that σ_{noise}=50uV, the offset seems so close to 0 it's not worth fitting, and I wanted to constrain p_{stim} to be 1. Now, the program came up with the following fit:


Quantal Analysis
FITTING EXPERIMENTAL DATA WITH A SIMPLE BINOMIAL DISTRIBUTION
Data amplitude histogram
Fitted probability distribution
Resample
Export fit
Data cumulative distribution
Fitted cumulative distribution
Plot Options
erase fit
Fit Options
Fit noise...
Fit
Test
Exit

Data directory: C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles

Data file: 133.amp

Resample: 10

No. bins: 50
Linewidth: 1
Fontsize: 8

No. starts: 10
Max. n: 10
Max. evals.: 1e+06

Fit Constraints:

		Fitted values
Number of release sites	☐ unconstrained	2
Release probability	☐ unconstrained	0.462
Quantal size Q	☐ unconstrained	317.209
Quantal variance	☐ unconstrained	73
Type of quantal variance	☐ unconstrained	flat

			Fitted values
Noise	✓ constrained	50	50.000
Conductance prob.	✓ constrained	1	1.000
Offset	✓ constrained	0	0.000

Fit type = MLE; glitch rate = 0.00

log(L) for current best fit = 1742.4434

Testing your fit

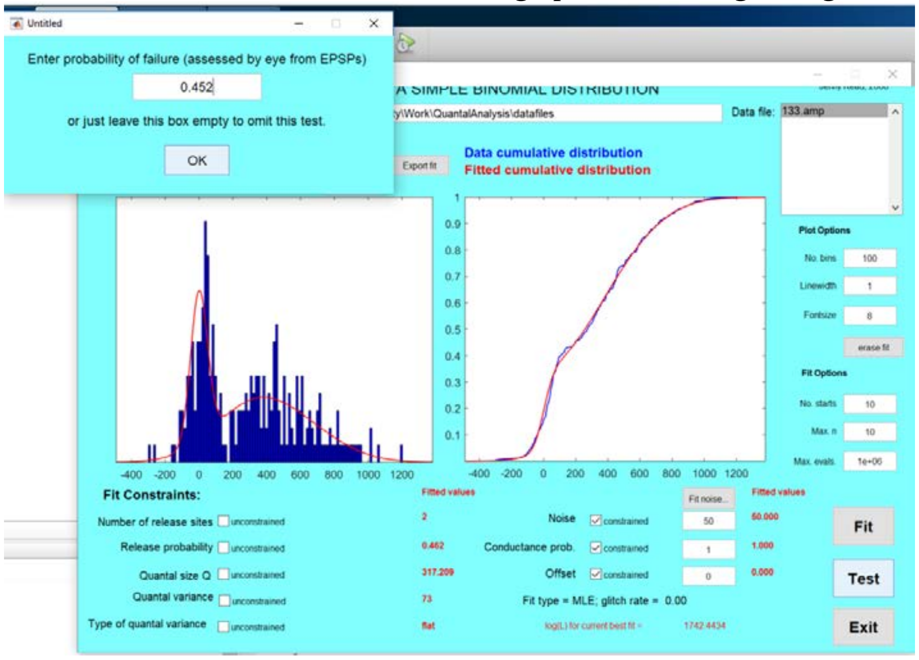
How do you know your fit bears any resemblance to reality? The short answer is you don't but my code provides two ways for you to do a sanity check.

Adequacy

Is it possible that the observed data came from the given model? The model CDF never matches the experimental one exactly, but are the discrepancies reasonable given the amount of data? (ie your statistical power to detect a difference)

This is described in Hardingham et al (2006) as follows:
"*Adequacy of fitted model*. To test whether the proposed fit was acceptable as a model of the experimental data, seven goodness-of-fit statistics were considered: the sum of the squared differences between the model and data cumulative distributions, C, the Kolmogorov–Smirnov *D* statistic (Press et al., 1993), and the χ^2 statistic for five different bin sizes. The power of the χ^2 statistic depends strongly on the bin size used. With too few bins, the test is too coarse to catch local deviations of the data from the model predictions. Conversely, if too many bins are used, the number of data points falling in any one bin is small and subject to large sampling fluctuations, so the statistic again tolerates poor fits. The optimal number of bins depends on the data set. By using a range of different bin numbers (20, 30, 50, 75, and 100) for each data set, we ensured that each data set would be exposed to a rigorous test. The distributions of these statistics under the null hypothesis, that the experimental data had actually been drawn from the fitted model, were obtained by Monte Carlo simulation (implemented in MATLAB on a personal computer). Five thousand sets of simulated data, each the same size as the experimental data set, were generated from the fitted model, and the seven goodness-of-fit statistics were calculated for each simulated data set. For each statistic, we calculated what proportion (f) of simulated data sets yielded higher values of the statistic (indicating worse fits) than the experimental data. A value of f<5% means that the null hypothesis cannot be rejected at the 5% level on the basis of the statistic. Finally, we applied an additional test, using the proportion of events that failed to evoke a simulated EPSP, p_{fail}. The Monte Carlo distribution of failure rates could then be compared with the p_{fail} observed experimentally. The failure rate test is a two-tailed test, so the null hypothesis is accepted at the 5% level provided that the experimental p_{fail} lies in between the 2.5 and 97.5% quantiles of the Monte Carlo distribution."

To assess this, hit "Test". That will bring up the following dialogue box.


Untitled
A SIMPLE BINOMIAL DISTRIBUTION
Export fit
Data cumulative distribution
Fitted cumulative distribution
Plot Options
erase fit
Fit Options
Fit noise...
Fit
Test
Exit
OK

Enter probability of failure (assessed by eye from EPSPs)

0.452

or just leave this box empty to omit this test.

y\Work\QuantalAnalysis\datafiles   Data file: 133.amp

No. bins 100
Linewidth 1
Fontsize 8
No. starts 10
Max. n 10
Max. evals 1e+06

Fit Constraints:

		Fitted values				Fitted values
Number of release sites	☐ unconstrained	2	Noise	✓ constrained	50	50.000
Release probability	☐ unconstrained	0.462	Conductance prob.	✓ constrained	1	1.000
Quantal size Q	☐ unconstrained	317.209	Offset	✓ constrained	0	0.000
Quantal variance	☐ unconstrained	73				
Type of quantal variance	☐ unconstrained	flat				

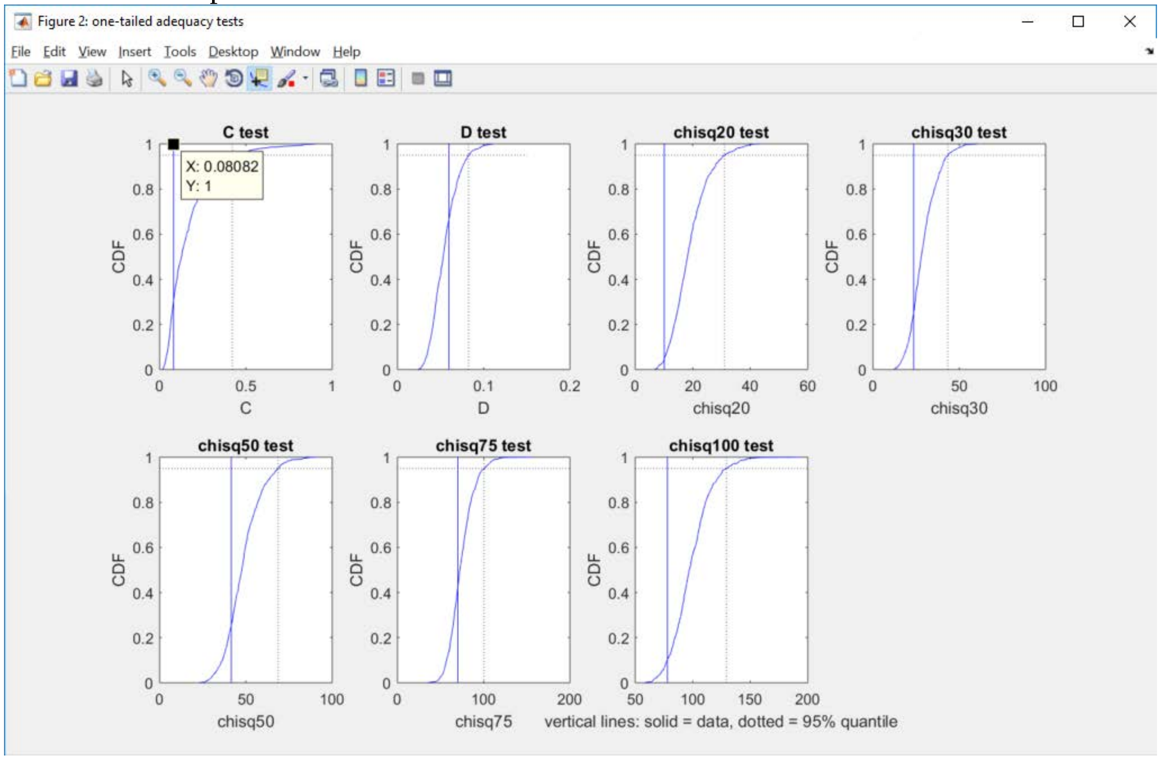
Fit type = MLE; glitch rate = 0.00

log(L) for current best fit = 1742.4434

What this is asking for is your estimate of the proportion of trials which failed to elicit a non-zero EPSP. Given the noise, this can't be gathered unambiguously from the raw data.

You might do something like, say, assume that all the EPSPs below the “notch” visible above at 200uV are actually failures. In the data set above, that gives you a proportion 0.452. You could then feed that into the dialogue box, and then the program will test that that proportion of failures could occur with the fitted model.

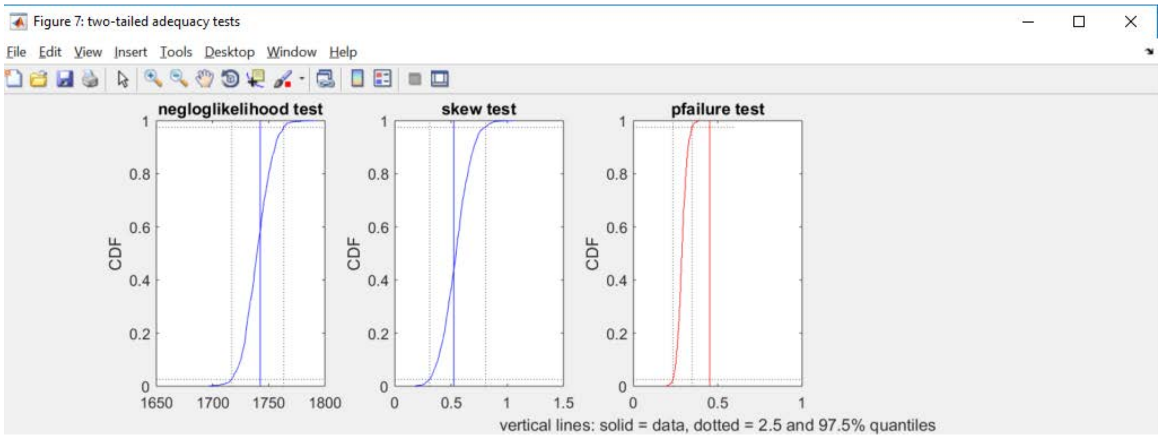
Here are the outputs for the one-sided test:



Hopefully this is clear given the explanation about. For example, C is the sum of the squared differences between the model and data cumulative distributions, summed over all EPSPs in the data file. This was 0.08082 for the actual data and the model fitted to the actual data (vertical blue line).
What we did now was generate 5000 sets of simulated data, using the probability density function f(v) of the fitted model. For each of these, we computed the C between that simulated data and the fitted model. You can see this varied between close to 0 and close to 1. 95% of these C were less than 0.4212 (vertical dotted line). Clearly the empirical C is well below this. This means that the model agrees with the empirical data as well as can be expected (at least as measured by C).

The other plots do the same thing for some other common test statistics, as described above. This model is clearly “adequate” as measured by the above one-sided tests, i.e. it cannot be rejected given the data.

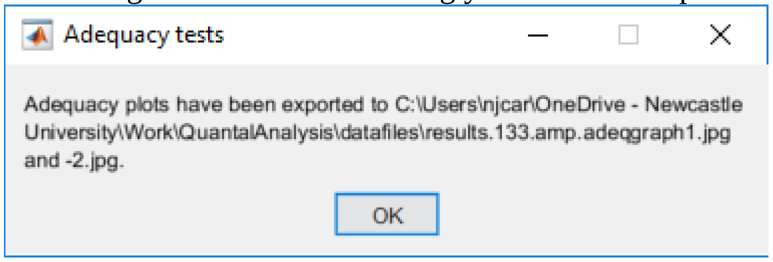
The second window is for the two-sided tests:



The first two of these weren’t discussed in Hardingham et al (2010). The first is just using –ln(L), the negative log likelihood, and the second the skew of the data. These just check that the empirical value of these quantities (vertical line) are also consistent with the range of values produced by the fitted model.

The third is pfailure, discussed above. Here, you can see that the model rejects the idea that the proportion of failures can be as high 0.452. The model predicts that the 95% confidence interval for pfail is 0.234 to 0.348, so 0.452 is grounds for rejecting the model. This would be a good reason to relax the constraint that p_{stim}=1 and allow the model to fit a lower value which would allow the model to predict higher pfailure.

You also get a little window telling you where these plots are saved, as jpgs:


Adequacy tests

Adequacy plots have been exported to C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles\results.133.amp.adeqgraph1.jpg and -2.jpg.

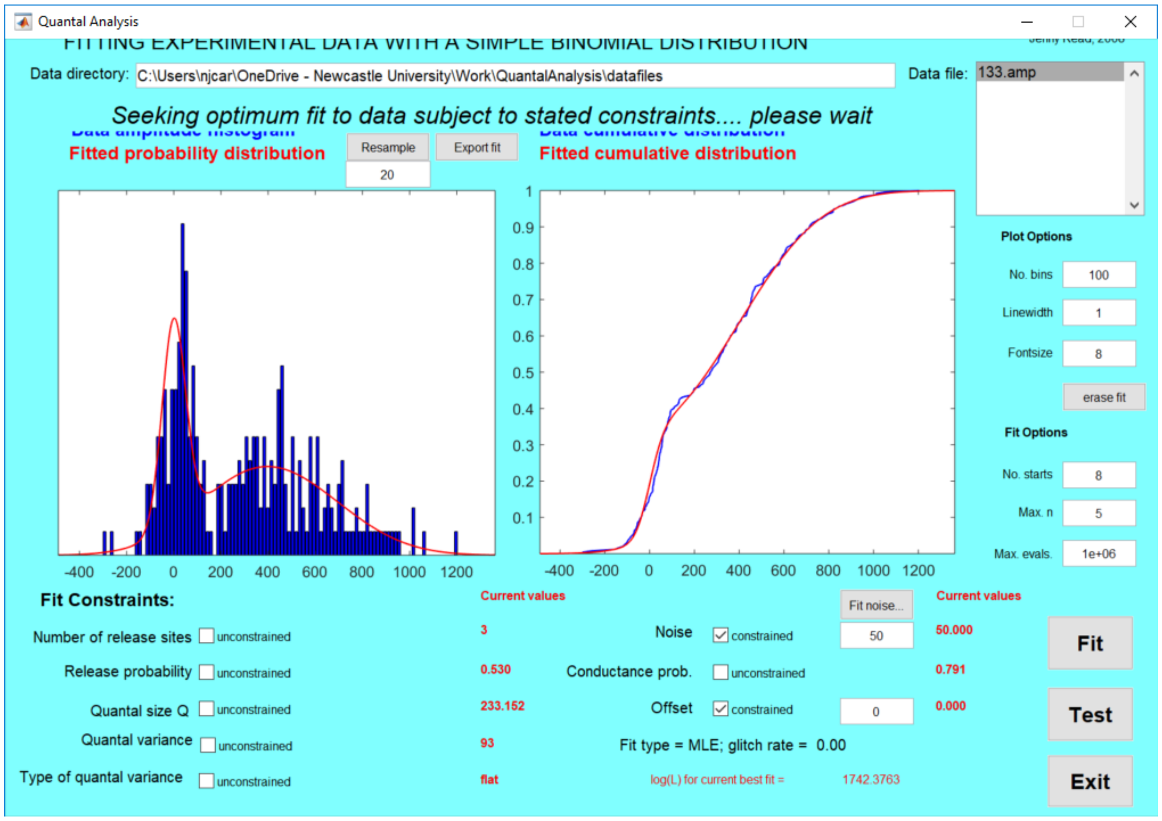
Uniqueness

The problem is of course that there are usually many many models which cannot be rejected given the data. This problem gets worse the less peaky the histograms are – if your histograms have lots of clear peaks, the parameters are probably well constrained; if they look like a mixture of two Gaussians, the parameters very much aren’t.

To assess this, hit “Resample” in the GUI. This will use bootstrap resampling as described in Hardingham et al (2010).

"*Bootstrap resampling.* Bootstrapping was used to estimate confidence intervals on fitted quantal parameters (Efron, 1979; Stricker et al., 1994). For each connection, we had *n* trials of recorded EPSPs. We generated new sets of data referring to sets of EPSP amplitudes for each connection by randomly selecting, with replacement, *n* EPSPs from the original set. Thus, some of the original EPSPs might appear more than once in the new set, whereas others might not appear at all. To avoid having several identical EPSPs in the new sets (which was correctly rejected by our battery of adequacy tests as being essentially impossible), we added a small amount of jitter to these resampled EPSPs. To each resampled EPSP, we added a random number drawn from a Gaussian, with a mean of 0 and an SD of one-quarter of the fitted noise SD (or five on the rare occasions in which one-quarter of the fitted noise SD was less than five). We then rounded the result to the nearest whole number to make all resampled EPSPs into integers, as they were in the original data (expressed as microvolts). This set of resampled EPSPs was then fitted in exactly the same manner as the original EPSPs were, and the resampled fit was tested for adequacy using the same selection of statistical tests that were applied to the original fit. If the resampled fit passed these tests, it was accepted as a valid resampled fit. The entire procedure was then repeated until 100 valid resampled fits had been acquired for each connection, giving 100 estimates of *Q*, *N*, and P_r for each connection"

100 resamples will take a long time to run. You can control how many resamples are done in the box in the GUI below "Resample". In the following example, I've reduced "No starts" to 8 and max n to 5, to speed things up.


Quantal Analysis

FITTING EXPERIMENTAL DATA WITH A SIMPLE BINOMIAL DISTRIBUTION

Data directory: C:\Users\njcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles

Data file: 133.amp

Seeking optimum fit to data subject to stated constraints.... please wait

Resample: 20

Plot Options
No. bins: 100
Linewidth: 1
Fontsize: 8

Fit Options
No. starts: 8
Max. n: 5
Max. evals: 1e+06

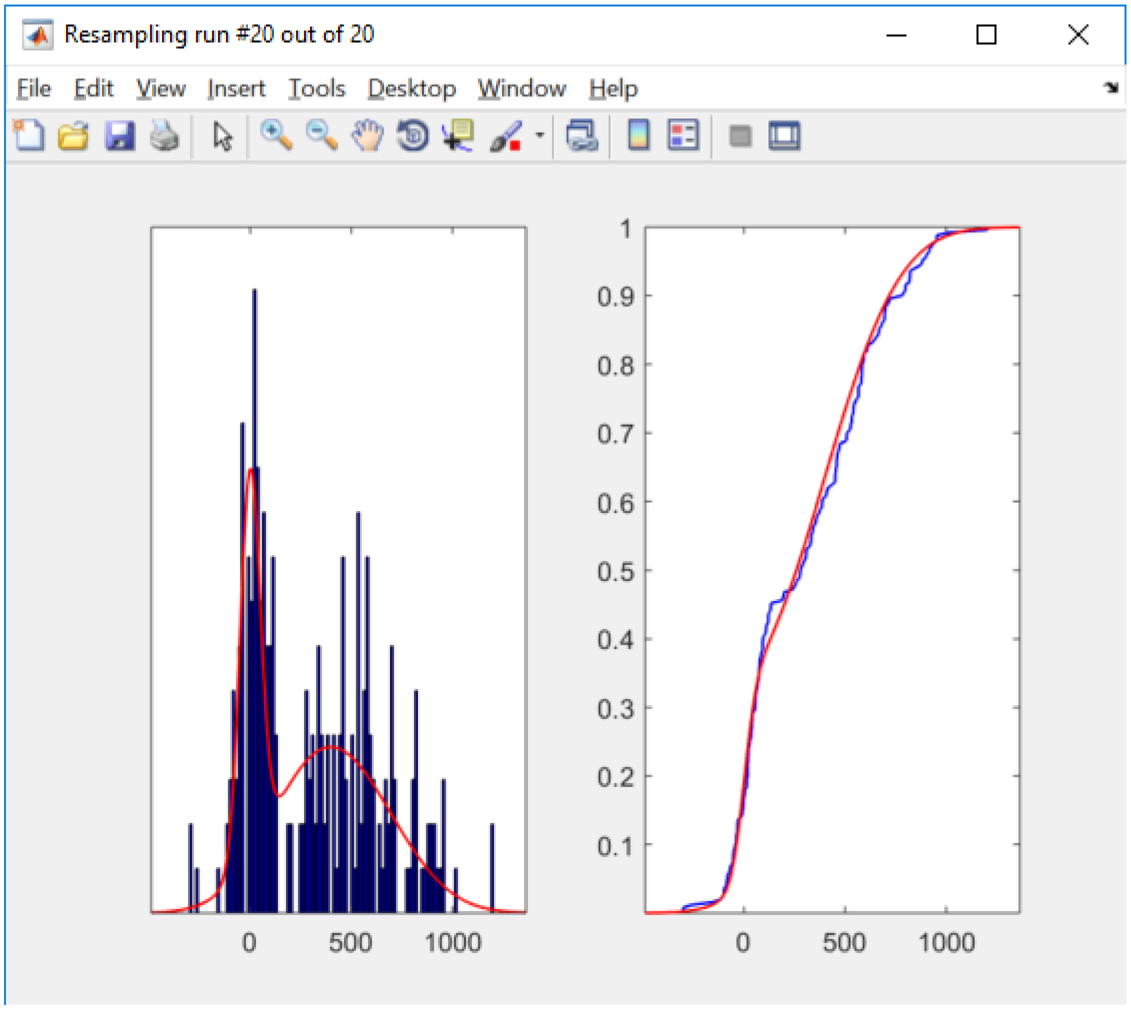
Fit Constraints:

		Current values				Current values
Number of release sites	☐ unconstrained	3	Noise	✓ constrained	50	50.000
Release probability	☐ unconstrained	0.530	Conductance prob.	☐ unconstrained		0.791
Quantal size Q	☐ unconstrained	233.152	Offset	✓ constrained	0	0.000
Quantal variance	☐ unconstrained	93				
Type of quantal variance	☐ unconstrained	flat				

Fit type = MLE; glitch rate = 0.00

log(L) for current best fit = 1742.3763

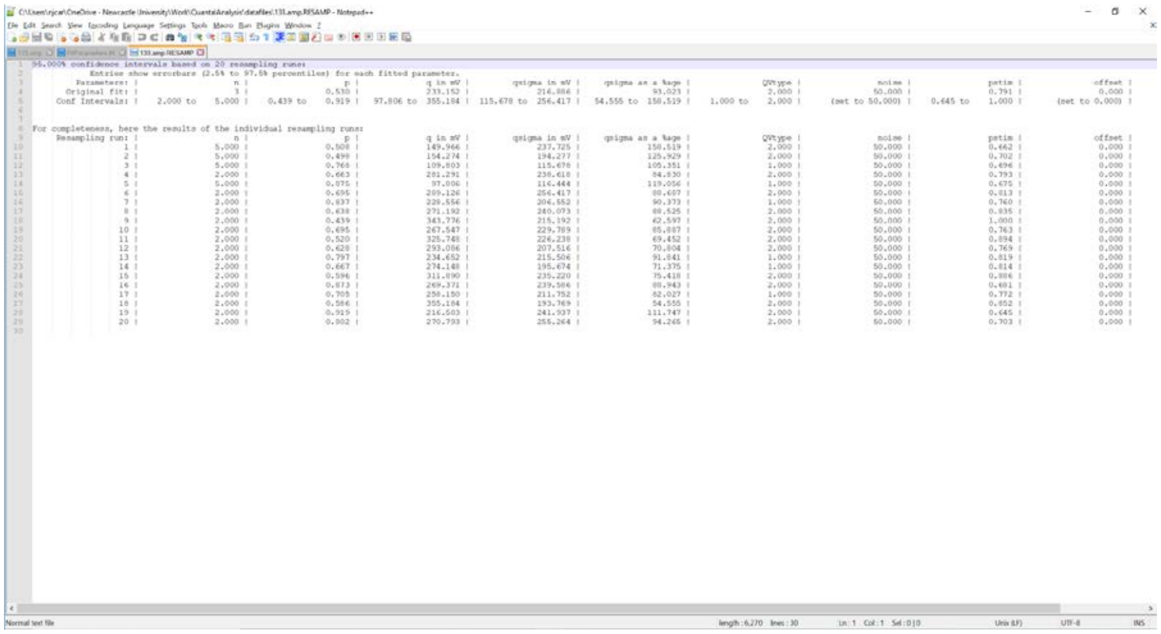
The program prints out example resampled data & fits as it goes:



You can see that the “data” is similar to but slightly different from the original experimental data in the main GUI.

The results are written out into the data directory (same filename as the data but with RESAMP at the end).

This is a text file so you can open it up in a text editor of your choice. It’s a wide file though so may not look good if the text wraps wron.
This is how it looks in Windows 10 Notepad++:



C:\Users\rjcar\OneDrive - Newcastle University\Work\QuantalAnalysis\datafiles\131.amp.RESAMP - Notepad++

```
95.000% confidence intervals based on 20 resampling runs:
            Entries show errorbars (2.5% to 97.5% percentiles) for each fitted parameter.
        Parameters: |                    n |                    p |                  q in mV |              qsigma in mV |        qsigma as a %age |                 QVtype |                 noise |                  pstim |                offset |
      Original fit: |                    3 |                0.530 |                  233.152 |                   216.886 |                  93.023 |                  2.000 |                50.000 |                  0.791 |                 0.000 |
    Conf Intervals: |    2.000 to    5.000 |    0.439 to    0.919 |   97.806 to    355.184 |  115.678 to       256.417 |  54.555 to      158.519 |    1.000 to      2.000 |     (set to 50.000) |    0.645 to      1.000 |       (set to 0.000) |


For completeness, here the results of the individual resampling runs:
      Resampling run: |                    n |                    p |                  q in mV |              qsigma in mV |        qsigma as a %age |                 QVtype |                 noise |                  pstim |                offset |
                   1 |                5.000 |                0.508 |                  149.966 |                   237.725 |                 158.519 |                  2.000 |                50.000 |                  0.662 |                 0.000 |
                   2 |                5.000 |                0.498 |                  154.274 |                   194.277 |                 125.929 |                  2.000 |                50.000 |                  0.702 |                 0.000 |
                   3 |                5.000 |                0.768 |                  109.803 |                   115.678 |                 105.351 |                  1.000 |                50.000 |                  0.696 |                 0.000 |
                   4 |                2.000 |                0.663 |                  281.291 |                   238.618 |                  84.830 |                  2.000 |                50.000 |                  0.793 |                 0.000 |
                   5 |                5.000 |                0.075 |                   97.006 |                   116.444 |                 119.056 |                  1.000 |                50.000 |                  0.675 |                 0.000 |
                   6 |                2.000 |                0.695 |                  209.126 |                   256.417 |                  88.607 |                  2.000 |                50.000 |                  0.813 |                 0.000 |
                   7 |                2.000 |                0.837 |                  228.556 |                   206.552 |                  90.373 |                  1.000 |                50.000 |                  0.760 |                 0.000 |
                   8 |                2.000 |                0.638 |                  271.192 |                   240.073 |                  88.525 |                  2.000 |                50.000 |                  0.835 |                 0.000 |
                   9 |                2.000 |                0.439 |                  343.776 |                   215.192 |                  62.597 |                  2.000 |                50.000 |                  1.000 |                 0.000 |
                  10 |                2.000 |                0.695 |                  267.547 |                   229.789 |                  85.887 |                  2.000 |                50.000 |                  0.763 |                 0.000 |
                  11 |                2.000 |                0.520 |                  325.748 |                   226.238 |                  69.452 |                  2.000 |                50.000 |                  0.894 |                 0.000 |
                  12 |                2.000 |                0.628 |                  293.086 |                   207.516 |                  70.804 |                  2.000 |                50.000 |                  0.769 |                 0.000 |
                  13 |                2.000 |                0.797 |                  234.652 |                   215.506 |                  91.841 |                  1.000 |                50.000 |                  0.819 |                 0.000 |
                  14 |                2.000 |                0.667 |                  274.148 |                   195.674 |                  71.375 |                  1.000 |                50.000 |                  0.814 |                 0.000 |
                  15 |                2.000 |                0.596 |                  311.890 |                   235.220 |                  75.418 |                  2.000 |                50.000 |                  0.886 |                 0.000 |
                  16 |                2.000 |                0.873 |                  269.371 |                   239.586 |                  88.943 |                  2.000 |                50.000 |                  0.681 |                 0.000 |
                  17 |                2.000 |                0.705 |                  258.150 |                   211.752 |                  82.027 |                  1.000 |                50.000 |                  0.772 |                 0.000 |
                  18 |                2.000 |                0.586 |                  355.184 |                   193.769 |                  54.555 |                  2.000 |                50.000 |                  0.852 |                 0.000 |
                  19 |                2.000 |                0.919 |                  216.503 |                   241.937 |                 111.747 |                  2.000 |                50.000 |                  0.645 |                 0.000 |
                  20 |                2.000 |                0.802 |                  270.793 |                   255.264 |                  94.265 |                  2.000 |                50.000 |                  0.703 |                 0.000 |
```

Normal text file   length : 6,270   lines : 30   Ln : 1   Col : 1   Sel : 0 | 0   Unix (LF)   UTF-8   INS

Hopefully this is self-explanatory. Obviously you don’t get confidence intervals for the parameters that were constrained, like offset and pstim in this example.
As noted in Hardingham et al (2010), “For most connections, these confidence intervals were large”. For example, above, p varies from 0.439 to 0.919.

Default parameters

Any changes you make in the GUI are written to FitParameters.jfit. This is just a text file. You can modify it manually if you like but you don’t need to. This just ensures that the next time you fire up the program, it starts with whatever you set the parameters to the last time, making it more convenient.