Problem 1: Fitting a simple counting experiment

Junior Lab Staff*
MIT Department of Physics
(Dated: February 24, 2009)

This solution includes sample distribution fits and iterative fits for method comparison.

1. DATA

The data we are given are from a counting experiment. The values are given in Table I and plotted in Figure 1. The uncertainties are taken as the square root of the number of events for a given count, as usual for counting experiments.

TABLE I: Raw data detailing how many observation events yielded a specified number of counts.

Counts	Events	Counts	Events	Counts	Events
0	5	5	36	10	1
1	10	6	22	11	2
2	24	7	14	12	4
3	40	8	3		
4	42	9	4		

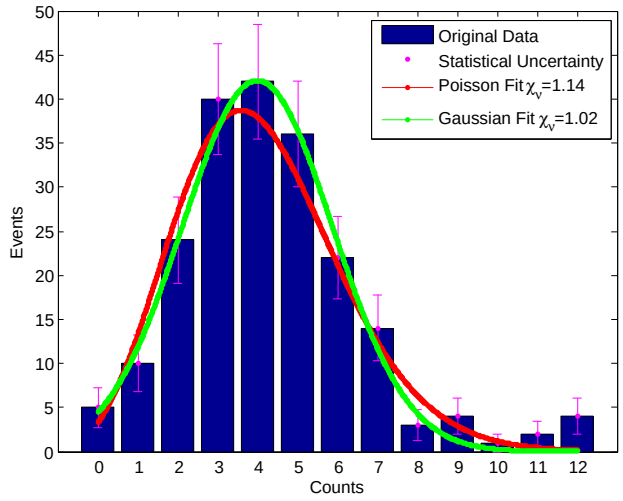


FIG. 1: Fitting exercise. Original data is shown in blue with magenta error bars. The iterative fits are shown in red and green.

*This solution was prepared by Daniel Nezich; URL: http://web.mit.edu/8.14/

2. FUNCTIONAL FORMS OF FITS

The two lineshapes we are asked to consider are the Poisson and the Gaussian distributions. The Poisson distribution is generated by considering that the occurrence of a count is equally probable at all times with a rate given by μ in units of counts per unit observation time (a special case for the Binomial distribution). The probability that an observation will produce k counts is given by

$$P(k,\mu) = \frac{\mu^k e^{-\mu}}{k!}\,. \tag{1}$$

It is worth noting that the standard deviation σ a Poisson distribution is equal to the root of its mean $\sigma = \sqrt{\mu}$.

The Gaussian distribution is also an approximation of the Binomial distribution, appropriate for large event rates μ (e.g. $\mu \geq 10$). Its standard deviation σ is an independent parameter.

$$P(k,\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(k-\mu)^2/2\sigma^2} \tag{2}$$

3. EXTRACTED PARAMETER FITS

The sample mean $\overline{x}$ and sample variance s^2 of the experimental data can be calculated and used in the above distribution functions under the reasonable approximations $\mu \approx \overline{x}$ and $\sigma \approx \sqrt{s^2}$. Designating count values by k_i and the corresponding number of events by n_i, the sample mean μ_{ex} is approximated by the average of the counts weighted by the number of events (Bevington[**?**] 4.9, see also Bevington 4.31)

$$\mu_{ex} = \frac{1}{N}\sum_{j=1}^{N} k_j = \frac{\sum_i k_i n_i}{\sum_i n_i} = 4.3285 \tag{3}$$

and the standard deviation σ is approximated by the square root of the sample variance (Bevington 4.13)

$$\sigma_{ex} = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}(k_j-\mu)} \approx \sqrt{\frac{\sum_i (k_i-\mu)^2 n_i}{\sum_i n_i}} = 2.2808\,. \tag{4}$$

Notice the difference between the two ways of expressing the sum. On the left, the summation is over all 207 events, whereas on the right the summation is over all 13 values of counts. The more generally correct method

of Bevington 4.31 is not used because statistical uncertainties have no effect on the result (the uncertainty in this problem is assumed to be strictly statistical). The uncertainty of the mean can be calculated as (Bevington 4.14, see also Bevington 4.31)

$$\sigma_{\mu,ex} = \frac{\sigma_{ex}}{\sqrt{N}} = \frac{2.2808}{\sqrt{207}} = 0.1585\,. \tag{5}$$

Note that the total number of events was used instead of the number of data points (13) because the number of data points is a result of binning our data, and that has no relevance to the statistics of the original data (although it will become important when we perform fits to the binned data, as described below). These parameters μ_{ex} and σ_{ex} can be used in the Poisson and Gaussian formulae 1 and 2. The goodness of fit can be characterized by the reduced chi-squared value

$$\chi^2_\nu = \frac{1}{\nu}\sum_i \left(\frac{k_i - f_i}{\sigma_i}\right)^2 \tag{6}$$

where the data points are the k_i and the fit points are the f_i. The uncertainties in the data points are used for σ_i. The parameter ν is the number of degrees of freedom in the data.

In this example, there are 13 data points, from which we subtract the number of constraints imposed by our fitting in order to obtain the number of degrees of freedom. For the Poisson distribution there are two constraints, the total number of events N and the mean number of counts μ. For the Gaussian distribution there are three constraints, N, μ, and the standard deviation σ. Note that for the Poisson distribution σ is not a constraint because it is trivially equivalent to the mean through $\mu = \sigma^2$. Thus the appropriate degrees of freedom are $13-2 = 11$ for a Poisson distribution and $13-3 = 10$ for a Gaussian distribution. The reduced chi-squared values are listed in Table III. For a discussion of the chi-squared value see section 4.4 of Bevington.

4. ITERATIVE FITS

Using the Matlab fitnonlin.m function with the total number of events, sample mean, and sample standard deviation listed in the top half of Table II as starting parameters, the values in the bottom half of Table II were calculated for Poisson and Gaussian functional forms. The specific Matlab code used to generate these fits appears at the end of this paper. The method employed by fitnonlin.m is the Levenberg-Marquardt algorithm detailed in section 8.4 and appendix E.4 of Bevington. The results of the fit are plotted in Figure 1 and the parameters are listed in Table II.

The fits can be evaluated by examining the chi-squared function discussed in the previous section. The results are listed in Table III. As reducedchi-squared values close to 1 are optimal, the Gaussian fit seems slightly better than the Poisson fit, which can be verified by visual inspection. By looking up the probability of observing these (or higher) reduced chi-squared values in Table C.4 of Bevington, we obtain Table III. If the parent distribution is Gaussian the data obtained is 42% likely, while if the parent distribution is Poisson the data obtained is 32% likely. As a metric, we can say that the Gaussian distribution is $42/(42+32) = 57\%$ likely over a Poisson distribution (which would be $32/(42+32) = 43\%$ likely).

TABLE II: Fitting parameters. Under the Distribution heading, P means Poisson, G means Gaussian, S means derived from the sample set, and F means obtained by fitting as described in the text. N is the number of events, μ is the mean number of counts, and σ is the standard deviation of the number of counts. The subscripted sigmas σ_m denote the uncertainty in parameter m (this is perhaps confusing and should be replaced by a δ notation, but is consistent with the treatment in Bevington.

Dist.	$N \pm \sigma_N$	$\mu \pm \sigma_\mu$	$\sigma \pm \sigma_\sigma$
PS	207 ± 14	4.33 ± 0.16	
GS	207 ± 14	4.33 ± 0.16	2.28 ± 0.51[a]
PF	194 ± 14	4.10 ± 0.13	
GF	198 ± 14	4.00 ± 0.14	1.88 ± 0.12

[a]Reference [**?**] gives the variance of the variance as $var(\sigma^2) = \sigma^2\sqrt{2/N}$ (Note, this needs to be verified). If this is so, then we have by the chain rule $\sigma_\sigma = \sigma_{\sigma^2}/2\sigma = \sigma\sqrt{2N}$. This result, however, has not been strictly derived, and is presented as an example of reasoning about uncertainties and to stimulate your further investigation.

TABLE III: Probabilities for finding an equal or larger reduced chi-squared value. The ideal case is 0.5 = 50%.

Dist.	χ_ν	ν	Prob.
PS	1.5314	11	0.1125
GS	2.8259	10	0.0016
PF	1.1449	11	0.3207
GF	1.0234	10	0.4202

5. MATLAB CODE

5.1. Fitting script

```
% Original data
counts = [0 1 2 3 4 5 6 7 8 9 10 11 12];
events = [5 10 24 40 42 36 22 14 3 4 1 2 4];

% Calculate uncertainty and total number of counts
uncertainty = sqrt(events);
N = sum(events);

% Calculate the sample mean, variance, and standard deviation
mean = sum(counts.*events)/N;
var = sum(((counts-mean).^2.*events))/(N-1);
std = sqrt(var);

% Poisson distribution fitting
p = pois(counts,[N,mean]);
[pf pfe pchi2f pfit] = ...
    fitnonlin(counts,events,uncertainty,'pois',[N,mean]);
pnu = (length(events)-2); % 2 Degrees of freedom
pchi2s = sum(((events-p)./uncertainty).^2);

% Gaussian distribution fitting
g = gaus(counts,[N,mean,std]);
[gf gfe gchi2f gfit] = ...
    fitnonlin(counts,events,uncertainty,'gaus',[N,mean,std]);
gnu = (length(events)-3); % 3 Degrees of freedom
gchi2s = sum(((events-g)./uncertainty).^2);

% Plot the results
bar(counts,events); hold on; ...
    errorbar(counts,events,uncertainty,'.m'); hold off;
X = [0:0.01:12];
hold on; plot(X,pois(X,pf),'.-r'); hold off;
hold on; plot(X,gaus(X,gf),'.-g'); hold off;

%Label the resulting graph
xlabel('Counts');
ylabel('Events');
% title('Preliminary Assignment I');
legend({'Original Data' 'Statistical Uncertainty' ...
    sprintf('Poisson Fit {\\chi_{\\nu}}=%3.2f',pchi2f/pnu) ...
    sprintf('Gaussian Fit {\\chi_{\\nu}}=%3.2f',gchi2f/gnu)});
axis([-1 13 0 50]);

% Calculate probablilities from chi-squared values
Pchi2nu = [1-chi2cdf(pchi2s,pnu);
1-chi2cdf(gchi2s,gnu);
1-chi2cdf(pchi2f,pnu);
1-chi2cdf(gchi2f,gnu)];
```

5.2. Poisson equation

```
function [ret] = pois(X,a)
A = a(1);
mu = a(2);
ret = A*(mu.^X.*exp(-mu))./gamma(X+1);
```

5.3. Gaussian equation

```
function [ret] = gaus(X,a)
A = a(1);
mu = a(2);
sigma = a(3);
ret = A/(sigma*sqrt(2*pi))*exp(-0.5*((X-mu)./sigma).^2);
```

Problem 2: Gaussian or Lorentzian?

Junior Lab Staff*
MIT Department of Physics
(Dated: February 24, 2009)

This problem investigates the difference between Lorentzian and Gaussian lineshapes, and the importance of considering the background during fitting. Sample distribution parameters and iterative fit parameters are given.

1. DATA

The two data files analyzed in this assignment are 'lineshape1.txt' and 'lineshapedata.txt', downloaded from the Junior Lab website Handouts section. The analysis technique and writeup, however, are still applicable.

The origin of the data is not specified, but the fact that all of the numbers are non-negative integers indicates that it is appropriate to assume that this is a counting experiment. For example, the data given could correspond to radiation detection events counted in a multichannel analyzer (with the bin number corresponding to energy), or to the digital output of a photometer (with the bin number related to the frequency of the light). Thus it is reasonable to assume Poisson statistics for the uncertainty in the measurements, as recommended in the problem statement.

2. FUNCTIONAL FORMS OF FITS

The two lineshapes we are asked to consider are the Gaussian and the Lorentzian distributions. The Gaussian distribution is derived as a limit of the Binomial distribution and is appropriate for sums of identically-distributed randomly chosen variables (review the Central Limit Theorem). The form of the Gaussian distribution is given below with amplitude A, mean μ, and standard deviation σ:

$$P(x,\mu,\sigma) = \frac{A}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{1}$$

The Lorentzian distribution is derived from the damped harmonic oscillator equation. It is thus applicable to many phenomena, from the amplitude of oscillation of a mass on a spring to the intensity of light emitted from an atom transitioning between electronic states. The Lorentzian distribution is markedly different from the Gaussian distribution in that it has infinite variance. For this reason, we use the full width at half maximum (FWHM) to describe the width of the Lorentzian distribution (the Gaussian has a FWHM of $2\sqrt{2\ln 2}\sigma$). The form of the Lorentzian distribution is given below with amplitude A, mean μ, and FWHM Γ:

$$P(x,\mu,\Gamma) = \frac{A}{\pi} \frac{\Gamma/2}{(x-\mu)^2 + (\Gamma/2)^2} \tag{2}$$

3. PROCEDURE

Using the Matlab fitnonlin.m function with the sample values as starting parameters, the fit parameters values were calculated for Gaussian and Lorentzian functional forms. The sample number N, sample mean $\overline{x}$, and sample variance s^2 of the experimental data were calculated for use in Equations 1 and 2 under the reasonable approximations $A \approx N$, $\mu \approx \overline{x}$, $\sigma \approx \sqrt{s^2}$, and $\Gamma \approx 2\sqrt{2\ln 2}\sqrt{s^2}$. These values are given in the first column of Table I, and are used as reasonable starting parameters for iterative fitting.

Note that a graphical origin for the starting parameters is also acceptable, and that the FWHM (which can be used directly in the Lorentzian distribution) should be converted to a standard deviation for use in the Gaussian distribution. An example of specific Matlab code used to generate these fits appears at the end of this paper. The method employed by fitnonlin.m is the gradient search algorithm detailed in section 8.4 and appendix E.4 of Bevington. The value of the variable chicut was changed to 10^{-7} to increase accuracy. The results of the fit are plotted along with their residuals, the fitting parameters are tabulated, and a conclusion is drawn regarding the appropriate lineshape.

4. DISCUSSION

4.1. Problem 2.1

The data in 'lineshape1.txt' was fit to a Gaussian and to a Lorentzian, as shown in Figure 1. The residuals are also shown. The parameters resulting from the fit are shown in Table I. The reduced-chi-squared values are very large, so that the probability of these values resulting from the parent distribution described by the fit parameters is very small ($< 10^{-13}\%$; this value seems formidably small, so one must understand that it is intended to represent ideal statistical error, and that if this assumption fails then these probabilities have little meaning). This, along with the even vertical distribution of

*This solution was prepared by Daniel Nezich; URL: http://web.mit.edu/8.14/

residuals for the Lorentzian fit (which show that the fit is qualitatively good because the data are everywhere centered around zero and randomly distributed), indicate that there is something wrong with the data. Most likely the uncertainty is larger than the Poisson values we assumed, incorporating experimental error in addition to statistical uncertainty. In any case, it is clear from comparison of residuals and reduced chi-squared values that the Lorentzian lineshape is the more appropriate for this data set.

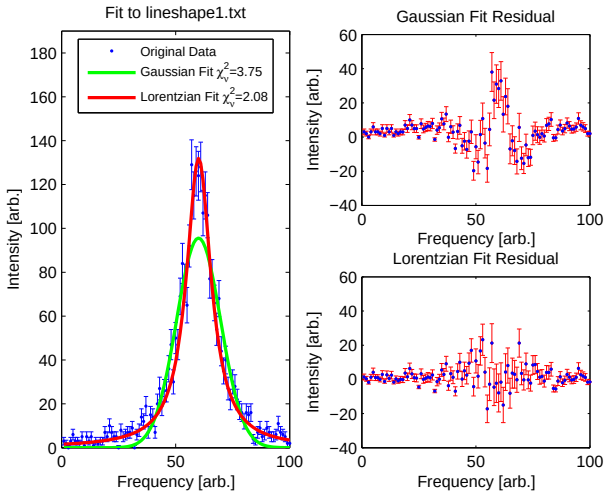


FIG. 1: Fit results for Problem 2.1.

TABLE I: Fitting parameters for lineshape1.txt data. Parameters are as described in the text. The uncertainties are given at the 68% level (one-sigma), and the number of degrees of freedom for the fitting is $100 - 3 = 97$.

	Sample	Gaussian Fit	Lorentzian Fit
N	2680 ± 52	2318 ± 48	2725 ± 55
μ	58.77 ± 0.28	60.06 ± 0.20	60.12 ± 0.19
σ	14.46 ± 0.08	9.67 ± 0.22	N/A
Γ	34.05 ± 0.19	22.78 ± 0.52	13.14 ± 0.37
χ^2_ν		3.75	2.08
$P(\chi^2_\nu)$		$< 10^{-13}$%	$2.4 \cdot 10^{-7}$%

One might think it is possible to improve the fit by adding an additional linear offset $(m * x + b)$ to the fitting equations. As can be seen by comparison of Table II with Table I, the factors added are small compared with their uncertainty, and do not significantly improve the reduced chi-squared value for the Lorentzian best fit. The reduced chi-squared value of the Gaussian fit is improved, but remains inferior to the Lorentzian fit; visual inspection suffices to show that trends are still present in the residuals of the Gaussian fit. Thus the addition of the linear offset terms do not seem justified for this data set.

One may also consider how robust the fitting method is by varying the initial guess for the fit parameters and

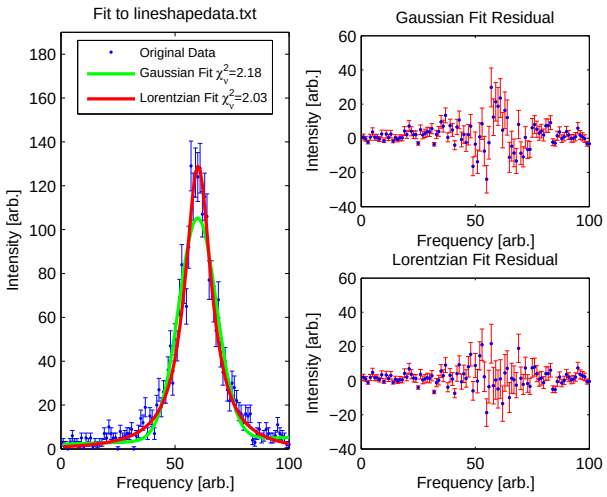


FIG. 2: Fit results for Problem 2.1 with linear baseline.

TABLE II: Linear background fitting results. Comparison with Table I shows the addition of the linear terms to the fit is not justified by improvement of fit quality as judged by χ^2_ν.

	Gaussian Fit	Lorentzian Fit
N	2109 ± 48	2880 ± 55
μ	59.91 ± 0.20	60.14 ± 0.19
σ	8.30 ± 0.20	N/A
Γ	19.56 ± 0.46	14.08 ± 0.38
b	2.22 ± 0.24	-0.82 ± 0.24
m	0.0302 ± 0.0049	-0.0070 ± 0.0049
χ^2_ν	2.18	2.03
$P(\chi^2_\nu)$	$2.6 \cdot 10^{-8}$%	$1.2 \cdot 10^{-6}$%

observing how the fitting results change. As an implementation of this, we take the fit parameter values from Table I and multiply them by $(1 + 0.25 * k_i)$ where $k_i \in \{-2, -1, 0, 1, 2\}$ (note that this method may be extended in a Monte Carlo manner to provide an even more convincing phase space coverage). We then perform a fitting for each of the 5^3 combinations of parameters N, μ, and σ/Γ thus generated (using them as initial values for the fit). The results are summarized in Table III, which give the mean and standard deviation of the 5^3 fit results. By comparison with the values in Table I we see that the variation due to changing the initial fit parameter guesses is negligible compared to the uncertainty derived from the data set. Thus we conclude that the gradient search algorithm is an adequately robust fitting method for this analysis.

4.2. Problem 2.2

In a similar manner as in Part I, the data in 'lineshapedata.txt' was fit to a Gaussian and to a Lorentzian. This

TABLE III: Robustness testing results. The mean and standard deviation of 125 fitting results starting from widely varying initial guesses for fit parameters are presented. Comparison with Table I shows that uncertainty due to initial guess is negligible.

	Gaussian Fit	Lorentzian Fit
N	2316.8 ± 1.8	2726.3 ± 2.0
μ	60.0599± 0.0008	60.1172 ± 0.0008
σ	9.666 ± 0.010	N/A
Γ	22.761 ± 0.023	13.133 ± 0.025
χ^2_ν	3.7531 ± 0.0001	2.0763 ± 0.0001
$P(\chi^2_\nu)$	$< 10^{-13}$%	$2.7 \cdot 10^{-7}$%

time, however, we consider three cases of fitting functions: 1) the usual Gaussian and Lorentzian distributions, 2) these distributions with an additional constant term, and 3) these distributions with a constant and a linear term. The initial guess for each fitting parameter is determined as in Part I (so that $N = 15860 \pm 130$, $\mu = 50.88 \pm 0.15$, $\sigma = 18.938 \pm 0.053$, and $\Gamma = 44.60 \pm 0.12$), except that the constant term is taken to be 50 and the linear coefficient is taken to be 1 (these values obtained by visual inspection of the data). The three cases for fitting are shown in Figures 3, 4, and 5 respectively. The parameters resulting from the fits are shown in Table IV and V for the Gaussian and Lorentzian functions respectively.

It is seen from the reduced chi-squared values that the Gaussian with a linear baseline is the most appropriate for this data set. The reduced chi-squared value is a little lower than desired, possibly indicating that the uncertainty in the data was overestimated. It is interesting to note that the quality of a fit whose function is missing a critical component will be limited by how well the portion of the function which is present approximates the missing component. In this example, when we consider the Lorentzian and Gaussian without any background, we find that the Lorentzian better approximates the linear component which is missing, by virtue of its long tails. Thus, without any background the Lorentzian fit is better than the Gaussian. This highlights the importance of using the correct functional form for your fit.

In concluding this exercise, let us again calculate the uncertainty due to the variation of our fitting function initial values. As was done in Part I, the mean and standard deviation the fit parameters which result from fitting with different initial values, are calculated and presented in Table VI. In this case, 3^5 points were chosen as before, with $k_i \in \{-2, 0, 2\}$. We see that the Junior Lab fitnonlin.m implementation is partially robust for our purposes by comparison with Tables IV and V. The uncertainties in N, μ, and σ/Γ are well below those arising from the data itself. The uncertainties in b and m, however, are larger than those from the data, and show that these variables are more susceptible to initial conditions, likely because they are a small adjustment to the fitting function. It might be appropriate to combine these errors, to perform fitting until convergence, or to seek another fitting method that is less sensitive to initial conditions.

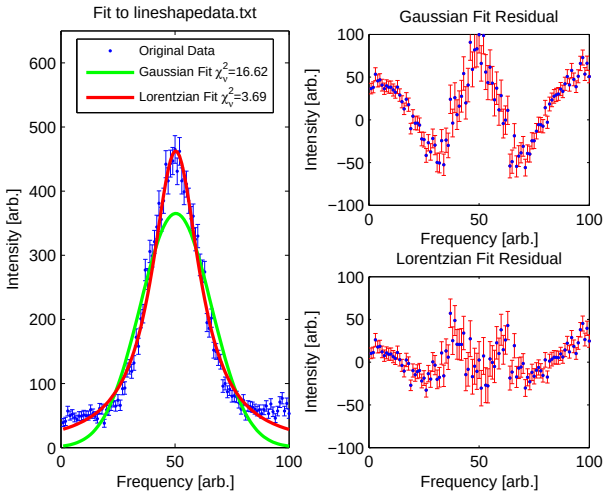


FIG. 3: Fit results for Problem 2.2: Gaussian and Lorentzian functions only.

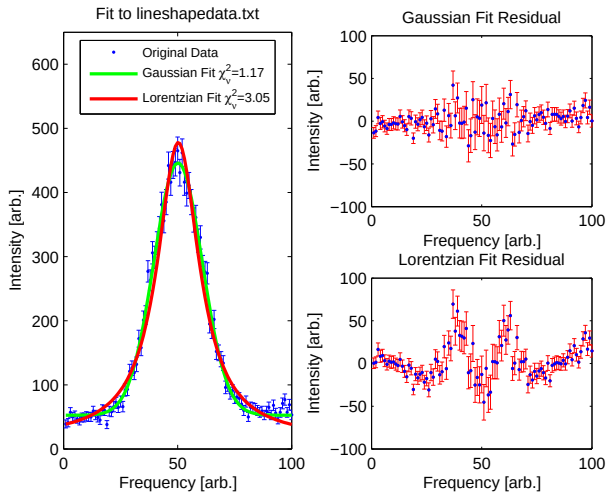


FIG. 4: Fit results for Problem 2.2: Gaussian and Lorentzian functions with constant baseline.

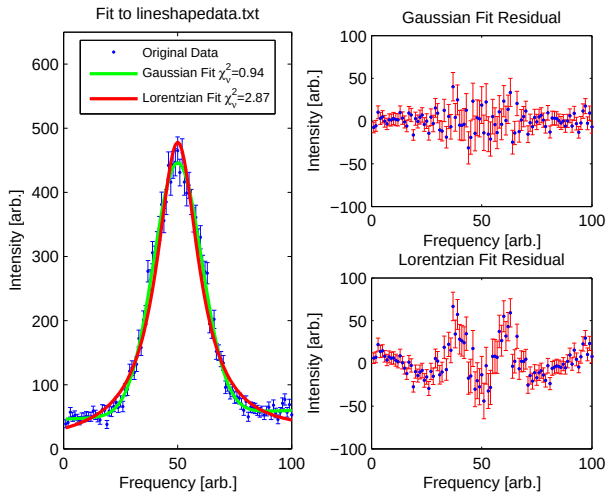


FIG. 5: Fit results for Problem 2.2: Gausssian and Lorentzian functions with linear baseline.

TABLE IV: Gaussian plus varying background fitting results. The column labels correspond to the type of background added to the Gaussian distribution.

	None	Constant	Linear
N	14270 ± 120	10480 ± 110	10460 ± 110
μ	50.36 ± 0.13	50.18 ± 0.13	50.01 ± 0.13
σ	15.58 ± 0.17	10.63 ± 0.11	10.61 ± 0.11
Γ	36.69 ± 0.41	25.03 ± 0.26	24.98 ± 0.26
b		52.64 ± 0.93	46.32 ± 0.93
m			0.135 ± 0.016
ν	97	96	95
χ^2_ν	16.6	1.17	0.940
$P(\chi^2_\nu)$	$< 10^{-13}$%	12.2%	64.6%

TABLE V: Lorentzian plus varying background fitting results. The column labels correspond to the type of background added to the Lorentzian distribution.

	None	Constant	Linear
N	18420 ± 150	16240 ± 140	16200 ± 140
μ	50.32 ± 0.13	50.30 ± 0.14	50.03 ± 0.14
σ	N/A	N/A	N/A
Γ	25.34 ± 0.31	22.37 ± 0.29	22.33 ± 0.29
b		16.06 ± 0.93	9.93 ± 0.93
m			0.131 ± 0.016
ν	97	96	95
χ^2_ν	3.69	3.05	2.87
$P(\chi^2_\nu)$	$< 10^{-13}$%	$< 10^{-13}$%	$< 10^{-13}$%

TABLE VI: Robustness testing results. The mean and standard deviation of 243 fitting results starting from widely varying initial guesses for fit parameters are presented. Comparison with Table IV and V shows that uncertainty due to initial guess is significant (on the order of the fit uncertainty) for the baseline parameters, but not for the curve parameters (which dominate by 1-2 orders of magnitude).

	Gaussian Fit	Lorentzian Fit
N	10456.2 ± 8.0	16260 ± 170
μ	50.018 ± 0.033	50.039 ± 0.050
σ	10.6067 ± 0.0070	N/A
Γ	24.977 ± 0.017	22.41 ± 0.22
b	46.8 ± 1.2	9.7 ± 1.9
m	0.127 ± 0.025	0.127 ± 0.024
χ^2_ν	0.949 ± 0.016	2.882 ± 0.018
$P(\chi^2_\nu)$	62% ± 9%	$< 10^{-13}$%

5. MATLAB CODE

5.1. Main File

```
% Original data and uncertianty
% Row 1 = Counts
% Row 2 = Number of events with the given count value
% Row 3 = Poisson uncertainty in number of events
%         (Poisson uncertainty of zero counts defined as 1)
data = dlmread('lineshapedata.txt');
data = [data; sqrt(data(2,:))];
data(3,find(data(3,:)==0)) = 1;

% Calculate total counts, sample statistics
N = sum(data(2,:));
mu = sum(data(1,:).*data(2,:))/N;
var = sum(((data(1,:)-mu).^2.*data(2,:)))/(N-1);
sigma = sqrt(var);

% Define some useful conversion functions
% (FWHM of Lorentzian is Gamma)
Gfwhm2std = inline('x/(2*sqrt(2*log(2)))');
Gstd2fwhm = inline('x*(2*sqrt(2*log(2)))');

% Define the functions we will fit to
gausfunc = inline(['a(1)/(a(3)*sqrt(2*pi))*'...
    'exp(-(1/2)*((x-a(2))./a(3)).^2)+a(4)+a(5)*x'],...
    'x','a');
lntzfunc = inline(...
    'a(1)*(a(3)/(2*pi))./((x-a(2)).^2+a(3)^2/4)+a(4)+a(5)*x',...
    'x','a');

% Fit to Gaussian with sample variables as initial values
ginitparam = [N,mu,sigma,50,1];
gnarg = length(ginitparam);
[gparam gunc gchi2 gfit] = ...
    fitnonlin(data(1,:),data(2,:),data(3,:),gausfunc,ginitparam);
gdof = length(gfit)-gnarg;
grchi2 = gchi2/gdof;

% Fit to Lorentzian with sample variables as initial values
linitparam = [N,mu,Gstd2fwhm(sigma),50,1];
lnarg = length(linitparam);
[lparam lunc lchi2 lfit] = ...
    fitnonlin(data(1,:),data(2,:),data(3,:),lntzfunc,linitparam);
ldof = length(lfit)-lnarg;
lrchi2 = lchi2/ldof;

% Fit to Gaussian repeatedly using a grid of initial values
ggrid = {}; ggridline = [0.5:0.5:1.5];
for k=1:gnarg, ggrid = [ggrid; {ggridline}]; end
garray = grideval(data, gausfunc, gparam, ggrid);
garray = shiftdim(garray,gnarg);
gmean = []; gstd = [];
for k=1:length(garray(:,1))
    gmean(k) = mean([garray{k,:}]);
    gstd(k) = std([garray{k,:}]);
end

% Fit to Lorentzian repeatedly using a grid of initial values
lgrid = {}; lgridline = [0.5:0.5:1.5];
for k=1:lnarg, lgrid = [lgrid; {lgridline}]; end
larray = grideval(data, lntzfunc, lparam, lgrid);
larray = shiftdim(larray,lnarg);
lmean = []; lstd = [];
for k=1:length(larray(:,1))
    lmean(k) = mean([larray{k,:}]);
    lstd(k) = std([larray{k,:}]);
end

% Plot the data and fits
subplot(2,2,[1 3]);
    errorbar(data(1,:),data(2,:),data(3,:),'.'); hold on;
    plot(data(1,:),gfit,'g',data(1,:),lfit,'r','Linewidth',2); hold off;
    xlabel('Frequency [arb.]');
    ylabel('Intensity [arb.]');
    title('Fit to lineshapedata.txt');
    legend({'Original Data' ...
        sprintf('Gaussian Fit {\\chi^2_{\\nu}}=%3.2f',grchi2) ...
        sprintf('Lorentzian Fit {\\chi^2_{\\nu}}=%3.2f',lrchi2)},...
        'Location','N',...
        'FontSize',8);
    axis([0 100 0 650]);
subplot(2,2,2);
    errorbar(data(1,:),data(2,:)-gfit,data(3,:),'.r'); hold on;
    plot(data(1,:),data(2,:)-gfit,'.b'); hold off;
    xlabel('Frequency [arb.]');
    ylabel('Intensity [arb.]');
    title('Gaussian Fit Residual');
    axis([0 100 -100 100]);
subplot(2,2,4);
    errorbar(data(1,:),data(2,:)-lfit,data(3,:),'.r'); hold on;
    plot(data(1,:),data(2,:)-lfit,'.b'); hold off;
    xlabel('Frequency [arb.]');
    ylabel('Intensity [arb.]');
    title('Lorentzian Fit Residual');
    axis([0 100 -100 100]);


% Calculate probablilities from chi-squared values
Pchi2nu = [...
    1-chi2cdf(gchi2,gdof);
    1-chi2cdf(lchi2,ldof)];
```

5.2. Initial Value Variation File

```
function [record] = grideval(data, func, base, grid, disp)
% Uses fitnonlin to find fit parameters starting from
% various inital values
% data:   Rows: 1)ordinate, 2)coordinate, 3)uncertainty
% func:   Name of the function to fit
% base:   Baseline initial values, same order as in func
% grid:   1-D array of cells containing 1-D arrays of
%         values to scale base by to obtain initial values
%         Index corresponds to variable index in base.
%         Empty cells are filled with the value 1
% disp:   (Optional) Outputs current grid values during fit
% record: Cell array of fitting results. The index in
%         record for a given result is the same as the
%         indices in grid of the initial values
%         (The final index is defined in line 53)
% Initialize the output
record = {};

% Handle dimensionality concerns
if length(base(:,1))~=length(grid)
    record = 'error: base and grid must have same size';
end
% Handle empty 'grid' values
for k=1:length(grid)
    if isempty(grid{k}), grid{k}=1; end
end
% Handle optional argument
if nargin<5
    disp = false;
end
```

```
% Determine if this is the basis case
basis = true;
for k=1:length(grid)
    if length(grid{k}) > 1
        basis = false;
        break;
    end
end

% Basis case: 'grid' has only singleton cells, perform evaluation
if(basis)
    % Generate initial guess for fit parameters
    initparam = base;
    for k=1:length(base)
        initparam(k) = initparam(k)*grid{k};
    end
    % Perform the fit
    [param err chi2 fit] = ...
        fitnonlin(data(1,:),data(2,:),data(3,:),func,initparam);
    dof = length(fit)-length(base);
    rchi2 = chi2/dof;
    record = {param err chi2 rchi2 fit}; % 1x5 result cells
    if disp, disp(initparam); end
    return
end

% Recursive case: evaluate along one nonsingleton dimension
% Find first nonsingleton dimension
indextoexpand = 0;
for k=1:length(grid)
    if length(grid{k})>1
        indextoexpand = k;
        break;
    end
end
% Initialize the results matrix
recdim = [];
for k=indextoexpand:length(grid)
    recdim = [recdim length(grid{k})];
end
record = cell([recdim 5]); % 1x5 result cells (see line 53)
% Iterate through the dimension and compile the results
for k=1:length(grid{indextoexpand})
    newgrid = grid;
    newgrid{indextoexpand} = grid{indextoexpand}(k);
    record(k,:) =...
        reshape(grideval(data, func, base, newgrid, disp),1,[]);
end
```

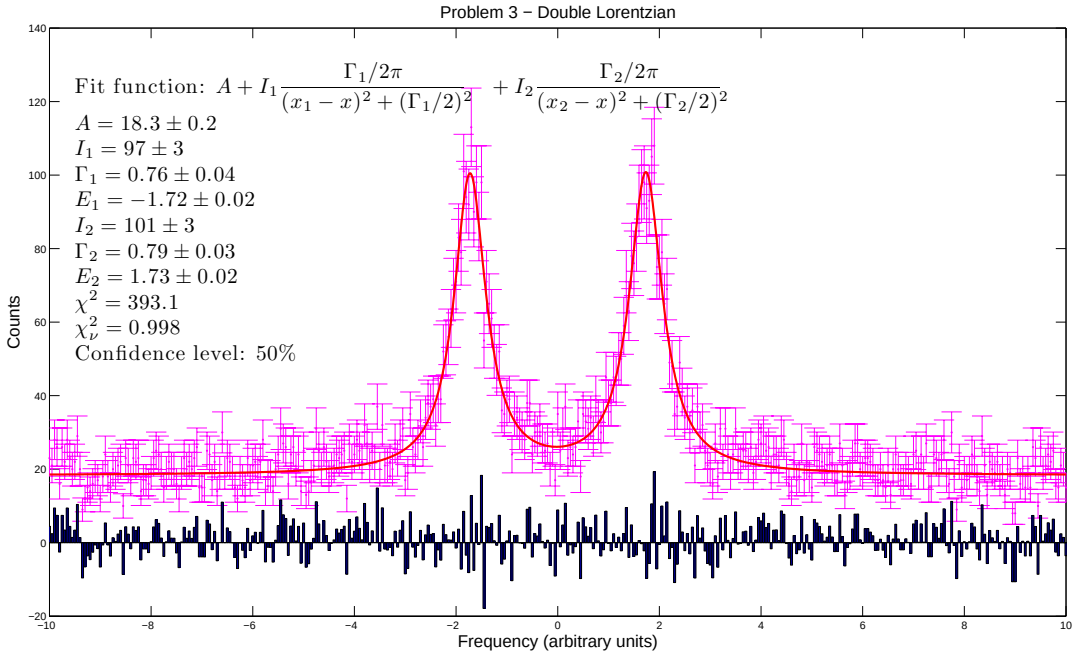
Problem 3: Fitting multiple peaks

Junior Lab Staff*
MIT Department of Physics
(Dated: March 4, 2009)

The data were fit to a sum of two Lorentzians plus a constant background using the gradsearch.m script posted on http://web.mit.edu/8.13/www/jlmatlab.shtml. The separation of the two peaks is given by the difference $\Delta E = |E_1 - E_2|$, and the uncertainty is obtained by summing the absolute errrors in quadrature:

$$\sigma^2_{\Delta E} = \sigma^2_{E_1} + \sigma^2_{E_2} \tag{1}$$

with the result $\Delta E = 3.46 \pm 0.03$.



Here is the matlab script used:

```
clear;
load dataex2.txt;
x = dataex2(1,:);
y = dataex2(2,:);
s = sqrt(1+y);

N=sum(y);

% First do the fit with a rough guess and a loose chicut parameter in gradsearch.m:
%guess = [N/2,-2,1,N/2,2,1,20];
%[a aerr chisq yfit] = fitnonlin(x,y,s,'dlor',guess);
```

*This solution was prepared by Steve Jaditz; URL: http://web.mit.edu/8.14/

```
% then put the resulting fitted parameters as the guess, and tighten chicut to 10e-10:
guess = [104.4399,-1.7274,0.9510,107.3026,1.7336,0.9335,17.5542];
[a aerr chisq yfit] = fitnonlin(x,y,s,'dlor',guess);

npar = length(guess);
confidence = 1-chi2cdf(chisq,length(y)-npar);
rChisq = chisq/(length(y)-npar);

deltaE = abs(a(5)-a(2))
sig_deltaE = deltaE*sqrt( (aerr(5)/a(5))^2 + (aerr(2)/a(2))^2 )

figure;
set(gcf,'Color','w');
errorbar(x,y,s,'.m');
hold on;
X = [x(1):0.01:x(length(x))];
plot(X,dlor(X,a),'-r','LineWidth',2);
bar(x,y-dlor(x,a));
v = axis;
axis([-10 10 v(3) v(4)]);
xlabel('Frequency (arbitrary units)','FontSize',16);
ylabel('Counts','FontSize',16);
title('Problem 3 - Double Lorentzian','FontSize',16);
eq = strcat('Fit function: $$A + I_{1} \frac{\Gamma_{1}/2\pi}{(x_{1} - x)^2 +',...
        '(\Gamma_{1}/2)^2}\quad +  I_{2}\frac{\Gamma_{2}/2\pi}{(x_{2} - x)^2 +',...
        '(\Gamma_{2}/2)^2}$$');
A = strcat('$$A = ',num2str(a(7),'%6.1f'),'\pm',num2str(aerr(7),'%-2.1f$$'));
I1 = strcat('$$I_1 = ',num2str(a(1),'%6.0f'),'\pm',num2str(aerr(1),'%-2.0f'),'$$');
G1 = strcat('$$\Gamma_1 = ',num2str(a(3),'%6.2f'),'\pm',num2str(aerr(3),'%-3.2f'),'$$');
E1 = strcat('$$E_1 = ',num2str(a(2),'%6.2f'),'\pm',num2str(aerr(2),'%-3.2f'),'$$');
I2 = strcat('$$I_2 = ',num2str(a(4),'%6.0f'),'\pm',num2str(aerr(4),'%-2.0f'),'$$');
G2 = strcat('$$\Gamma_2 = ',num2str(a(6),'%6.2f'),'\pm',num2str(aerr(6),'%-3.2f'),'$$');
E2 = strcat('$$E_2 = ',num2str(a(5),'%6.2f'),'\pm',num2str(aerr(5),'%-3.2f'),'$$');
s_chisq = strcat('$$\chi^2 = ',num2str(chisq,'%6.1f'),'$$');
s_rChisq= strcat('$$\chi_{\nu}^2 = ',num2str(rChisq,'%6.3f'),'$$');
s_confidence   = strcat('Confidence level: $$',num2str(confidence*100,'%6.0f'),'\%$$');
label = {eq;A;I1;G1;E1;I2;G2;E2;s_chisq;s_rChisq;s_confidence};
text('Position',[-9.5 90],'Interpreter','latex','FontSize',20,'String',label);
hold off;
```

And the function dlor.m:

```
function [ret] = lor(X,a)
     A = a(1);
     E = a(2);
     G = a(3);
     A2= a(4);
     E2= a(5);
     G2= a(6);
     B = a(7);
     ret = B + A*G/2/pi./((E-X).^2+(G/2)^2) + A2*G2/2/pi./((E2-X).^2+(G2/2)^2);
```