Investigating Data for Radioactive Decay of a Substance

Ben Tate[redacted]

Submitted: 20/12/2022

An investigation was conducted to analyse the radioactive decay of a substance. Two data sets were given in the form of activity recorded by 4 detectors at different distances at given time intervals. From analysis of the first dataset, it was determined the substance had a half-life of $17.97\text{s}^{+0.059}_{-0.060}$ and was emitted isotopically from a source of 3,302,595 ± 108,114 particles. This strongly suggests the first substance was ^{139}Eu. The second dataset was discovered to be a mixture of two isotopes, likely ^{139}Eu and ^{168}W, also emitted isotopically from 17,781,122 ± 143,167 particles.

1. Introduction

There are various different analytical methods used throughout the investigation, but the simplest and possibly most useful is the Weighted Least Squares fitting. This method relies on minimising the distance between the data points and a linear model, at each point. It can be solved for any linear dataset using the **Equations 1** and **2** for 'm' and 'c' below.

$$m = \frac{S\,S_{xy} - S_x S_y}{S\,S_{xx} - S_x^2} \quad (1)$$

$$c = \frac{S_{xx} S_y - S_x S_{xy}}{S\,S_{xx} - S_x^2} \quad (2)$$

With:

$$S = \sum_i \frac{1}{\sigma_i^2}$$

$$S_x = \sum_i \frac{x_i}{\sigma_i^2} \; S_y = \sum_i \frac{y_i}{\sigma_i^2}$$

$$S_{xx} = \sum_i \frac{x_i^2}{\sigma_i^2} \; S_{xy} = \sum_i \frac{x_i y_i}{\sigma_i^2}$$

Where m and c are the gradient and y-intercept of the model.

This is a very simply way of fitting a model to a linear data with its respective uncertainties. A code snippet of the Weighted Least Squares function used in this investigation can be found in **Appendix 1**.

We can also use numerical methods to estimate the uncertainties associated with each parameter found using the Weighted Least Squares function. A quick and easy way to do this is the Bootstrap method. This involves subtracting the best fit model from the data to get the residuals, the absolute distance between the model and each data point. Next residuals are drawn at random, with replacement, and added to the best fit model. A new best fit model is then calculated using the Weighted Least Squares method. This process is repeated a large number of times, and the parameters stored for each.

Finally, a histogram is plotted with every iteration of a given parameter. Median and 16% - 84% confidence intervals can then be calculated for each parameter thus giving the uncertainties. See **Appendix 2** for the Bootstrap function used in this investigation.

The Chi-Squared value is a single number which shows how modelled data compares to actual data. Minimising Chi-Squared is a good way to improve a model and find the best possible parameters. There are many ways to minimise the Chi-Squared statistic, however for this investigation a grid-based approach was used. This involved calculating the Chi-Squared value for each point on the grid to find the minimum. An example of the Chi-Squared function is found in **Appendix 3**.

Another important statistical process used in this investigation is the Markov Chain Monte Carlo (MCMC). This was used instead of the grid-based Chi-Squared approach for the second dataset, due to the large number of parameters and limited processing power and storage. The MCMC process starts at a given point in the parameter space. It then randomly selects a new point in the parameter space from a probability function centred on the point. The Chi-Squared value is calculated at this point and compared with the previous. If the Chi-Squared value is lower we jump to the new point, but if its higher, there is a probability of $e^{-\frac{\Delta X^2}{2}}$ to jump to the new position. This process is then repeated a very large number of times. Histograms can then be plotted to determine best fit parameters and their associated uncertainties. The MCMC function used can be found in **Appendix 4**

For the analysis in this report, the models were based off **Equation 3** below.

$$A = A_0 e^{-\lambda t} = N_0 \lambda e^{-\lambda t} \quad (3)$$

Where A is activity of sample, A_0 is the Activity at start, N_0 is the number of particles in the sample at the start, λ is the decay constant and t is the time.

This function was linearised in order to perform a Weighted Least Squares fit according to **Equation 4**.

$$\ln(A) = -\lambda t + \ln(A_0) \quad (4)$$

For the second data set, the possibility of 2 isotopes present was explored. The data was then modelled by **Equation 5** below.

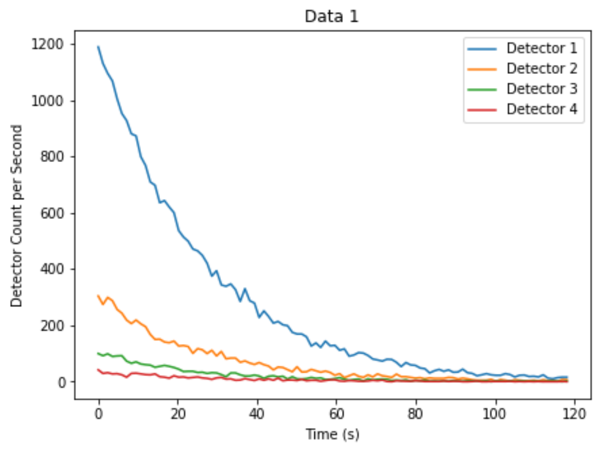
$$A = A_{0,1} e^{-\lambda_1 t} + A_{0,2} e^{-\lambda_2 t} \quad (5)$$

2. First Data Set

NOTE: This report only includes a selection of graphs and results to keep it brief. Full results can be found in Jupyter Notebook Assignment4.ipynb

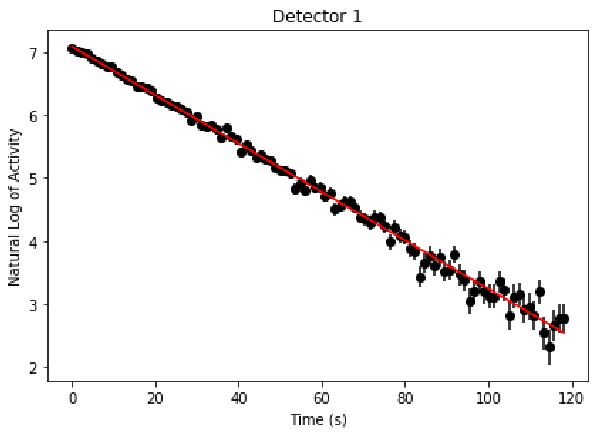
The data used in this investigation consisted of activities for given time intervals recorded by 4 detectors along with their associated uncertainties. These needed to be converted to activity per second before any analysis could be done. This was done by dividing the activity by the difference in time interval between 2 consecutive points.

Initially the data was plotted to get an idea of how the analysis should be carried out. This can be found in **Graph 1**.



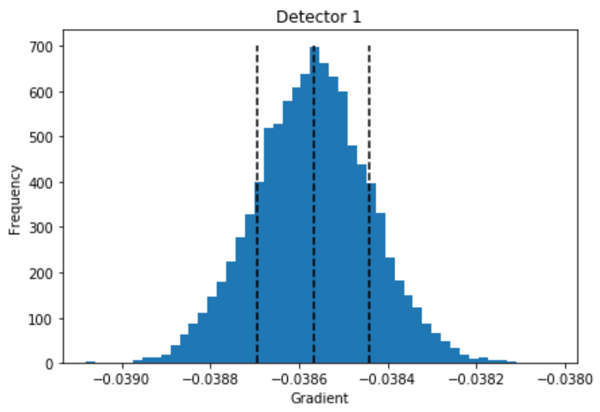
Graph 1: Activity per second against Time for the first dataset for each detector

As discussed in the introduction, to perform a Weighted Least Squares fitting, the data needs to be linear. I did this by taking the natural log of the Activity per second as in **Equation 4**. The best fit model was calculated for each detector and plotted along with the natural log data points and their propagated error bars. The plot for detector 1 can is seen in **Graph 2**.



Graph 2: Natural Log of Activity per second against Time for the first dataset for detector 1

Next the residuals were calculated, and a Bootstrap analysis was performed to determine the uncertainties associated with each parameter in the least squares fitting. One of the resulting histograms can be seen in **Histogram 1**.



Histogram 1: Frequency against gradient parameter for detector 1 from the first dataset, with median, upper and lower Confidence Intervals

The parameters and their uncertainties were then transformed back into their physical forms of decay constant and A_0. Half life is a more useful measurement than decay constant and can be calculated according to **Equation 6**.

$$t_{\frac{1}{2}} = \frac{\ln(2)}{\lambda} \quad (6)$$

You can find half-life, decay constant and A_0 values for each detector along with their uncertainties in **Table 1, 2** and **3** respectively.

	Half-Life (s)	**Error (s)**
Detector 1	17.972	+0.058 -0.060
Detector 2	18.209	+0.164 -0.164
Detector 3	18.225	+0.234 -0.225
Detector 4	20.527	+0.541 -0.548

Table 1: Calculated Half-Life and uncertainties for each detector in the first dataset, from Weighted Least Squares and Bootstrap

	Decay Constant (s^{-1})	**Error (s^{-1})**
Detector 1	0.03857	+0.00012 -0.00013
Detector 2	0.03807	+0.00034 -0.00034
Detector 3	0.03803	+0.00047 -0.00047
Detector 4	0.03377	+0.00089 -0.00090

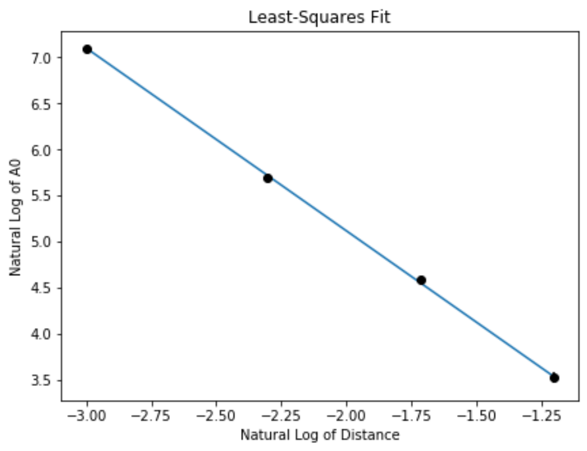
Table 2: Calculated Decay Constant and uncertainties for each detector in the first dataset, from Weighted Least Squares and Bootstrap

	A_0 (Bq)	**Error (Bq)**
Detector 1	1206.1	+10.6 -10.3
Detector 2	296.2	+6.7 -7.1
Detector 3	98.1	+3.0 -3.2
Detector 4	34.1	+2.1 -2.2

Table 3: Calculated A_0 and uncertainties for each detector in the first dataset, from Weighted Least Squares and Bootstrap

Using the Half-Life values from **Table 1** I can confidently say that the sample consists of ^{139}Eu atoms. The value calculated from detector 1 has by far the smallest uncertainty and therefore that is the reading I have based this conclusion on.

The next part of the investigation involved determining whether the radiation was isotropic. The values of A_0 were used along with the distances of each detector from the source. The natural log of these values were calculated so a linear relationship could be established. Once again Weighted Least Squares and Bootstrap analysis were performed to determine the parameters and uncertainties of the model. A plot of the model and the data can be seen in **Graph 3**.



Graph 3: Natural Log of Activity at start against Natural Log of Distance (m) for the first dataset

The Powerlaw index was calculated as ≈ -2 suggesting that the source is indeed isotropic, and activity at a distance x follows the inverse square law. The results and uncertainties can be found in **Table 4**.

	Value	Error
Powerlaw Index	-1.983	+0.018 -0.019
Powerlaw Constant	3.163	+0.116 -0.124

Table 4: Calculated Powerlaw Index and Constant for the first dataset

The final stage of the investigation for the first dataset required a comparison of the results in **Tables 2** and **3** with those found using the Python SciPy curve_fit function. These are found in **Tables 5** and **6,** and are very similar to previous calculated values. This suggests both methods are good and the results are accurate.

	Decay Constant (s^{-1})	+/- Error (s^{-1})
Detector 1	0.03889	0.00023
Detector 2	0.03986	0.00057
Detector 3	0.04148	0.00086
Detector 4	0.03856	0.00148

Table 5: SciPy curve_fit Decay Constant and uncertainties for each detector in the first dataset

	A_0 (Bq)	+/- Error (Bq)
Detector 1	1210.7	9.3
Detector 2	302.2	5.5
Detector 3	101.4	2.7
Detector 4	33.9	1.5

Table 6: SciPy curve_fit A_0 and uncertainties for each detector in the first dataset

3. Second Data Set

The analysis of the second dataset started in a similar way to the first. Weighted Least Squares and Bootstrap methods were used to determine the parameters and thus decay constant and A_0 values for the data with their uncertainties. These are found in **Tables 7** and **8**.

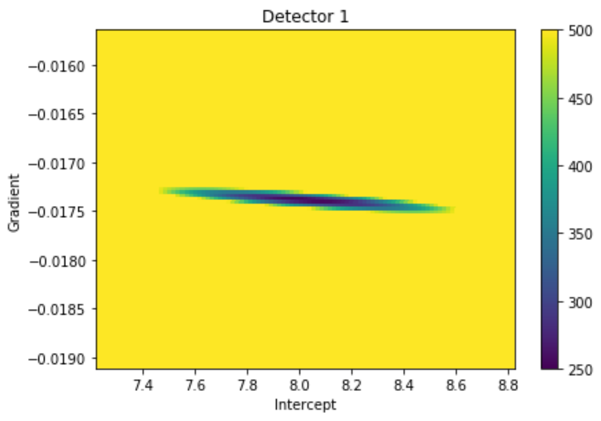
	Decay Constant (s^{-1})	Error (s^{-1})
Detector 1	0.01738	+0.00013 -0.00013
Detector 2	0.01715	+0.00017 -0.00018
Detector 3	0.01754	+0.00029 -0.00029
Detector 4	0.01730	+0.00050 -0.00048

Table 7: Calculated Decay Constant and uncertainties for each detector in the second dataset, from Weighted Least Squares and Bootstrap

	A_0 (Bq)	Error (Bq)
Detector 1	3051.4	+26.7 -26.4
Detector 2	756.7	+9.0 -9.2
Detector 3	240.4	+4.6 -4.9
Detector 4	85.9	+2.8 -3.0

Table 8: Calculated A_0 and uncertainties for each detector in the second dataset, from Weighted Least Squares and Bootstrap

An alternative way of ensuring these parameters are optimal for this model is to perform a Chi-Squared analysis. A grid-based approach was taken, to determine what parameter values gave the lowest Chi-Squared values. An example of the Chi-Squared heatmap for detector one of the second dataset can be found in **Graph 4**.



Graph 4: Heatmap of Chi-Squared for grid of best fit parameters from Weighted Least Squares for detector 1 of second dataset

The minimum Chi-Squared values for each detector along with the parameters which they occur at is located in **Table 9**.

	Minimum χ^2	Decay Constant (s^{-1})	A_0 (Bq)
Detector 1	255.6	0.01750	3076.8
Detector 2	144.3	0.01723	761.8
Detector 3	110.2	0.01763	241.7
Detector 4	112.8	0.01724	85.5

Table 9: Minimum Chi-Squared values and their associated parameter values

Although Chi-Squared is a very useful tool when looking at models, on its own it is not well suited for model selection partially due to the risk of overfitting. Taking the reduced Chi-Squared can solve this, however there are two Information Criterions that are better suited for model selection.

These are known as the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), found in **Equations 7** and **8** respectively.

$$AIC = \chi^2 + 2p + \frac{2p(p+1)}{N-p-1} \quad (7)$$

$$BIC = \chi^2 + \ln(N)\,p \quad (8)$$

Where p is the number of parameters in the model and N is the number of data points

The python function used to calculate these for this investigation is found in **Appendix 5**. The values output by the function were broadly similar to the Chi-Squared values and are located in **Table 10**.

	Minimum χ^2	**AIC**	**BIC**
Detector 1	255.6	259.7	264.8
Detector 2	144.3	147.7	152.8
Detector 3	110.2	114.2	119.3
Detector 4	112.8	116.9	122.0

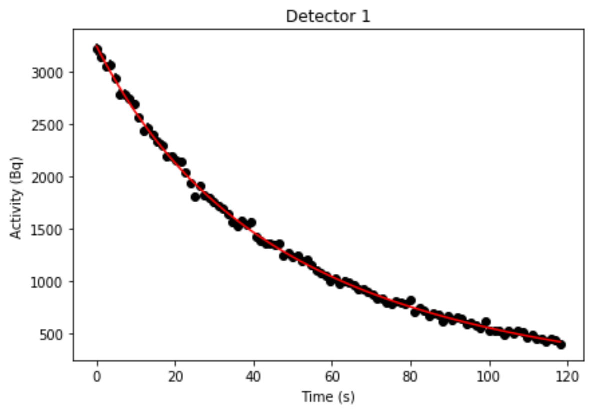
Table 10: Minimum Chi-Squared values and their associated AIC and BIC values for a single isotope model

To further the investigation, it was assumed that the sample was made up of 2 isotopes, and **Equation 5** was used to model the data. For ease of use and the fact it performed very similar to the Weighted Least Squares fit for the first data set, SciPy curve_fit function was used. The parameters output are in **Table 11** below.

	Decay Constant (s^{-1})	**+/- Error (s^{-1})**	**A_0 (Bq)**	**+/- Error (Bq)**
	Isotope 1			
Detector 1	0.01449	0.00085	2322.6	250.2
Detector 2	0.01232	0.00322	442.6	215.3
Detector 3	0.01625	0.00069	214.7	11.5
Detector 4	0.01635	0.00103	76.1	5.6
	Isotope 2			
Detector 1	0.04259	0.00805	916.1	238.3
Detector 2	0.03359	0.01147	361.8	208.9
Detector 3	0.09737	0.04083	53.0	11.9
Detector 4	0.12006	0.07779	20.1	7.0

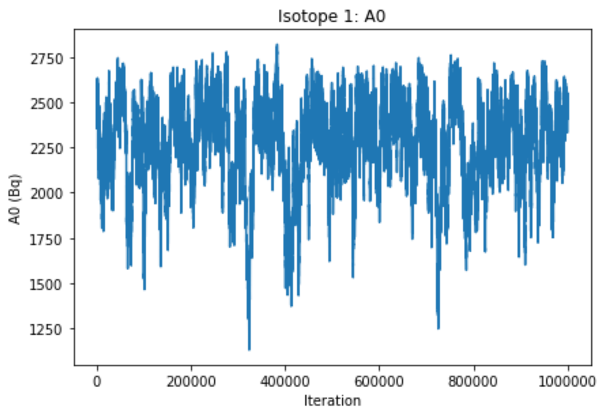
Table 11: Best fit parameters from curve_fit for a 2-isotope model, and associated errors.

The resulting models were plotted over the points with their error bars to ensure they were visually consistent. This is only a quick first check to make sure the model and parameters is sensible before moving on to more rigorous analysis. The plot for detector one is found in **Graph 5**.



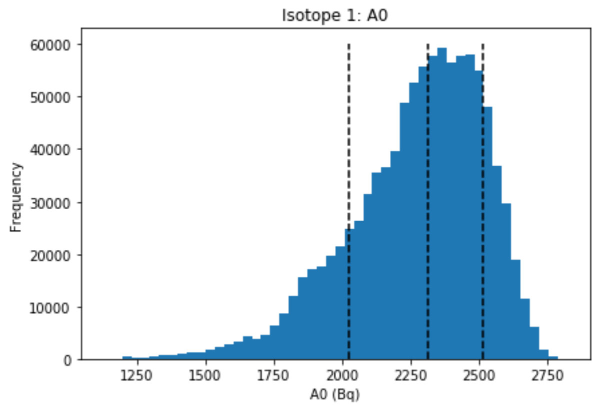
Graph 5: Activity per second against Time for the second dataset for detector 1 with a 2-isotope model overplot

Based on the best fit parameters from the curve_fit, the fitting was repeated with an MCMC to determine the final parameters and uncertainties, as well as the best fit Chi-Squared value. When running an MCMC it is important experiment with step sizes to ensure the jump fraction is adequate. The jump fractions for each detector in this investigation are between 12% and 19%, which is within the ideal range. Each of the MCMC chains were then plotted to check for any strange behaviour, and start positions adjusted accordingly. See **Graph 6** for the chain plot of the A_0 value of isotope 1, detector 1. Each MCMC was run for 1,000,000 iterations.



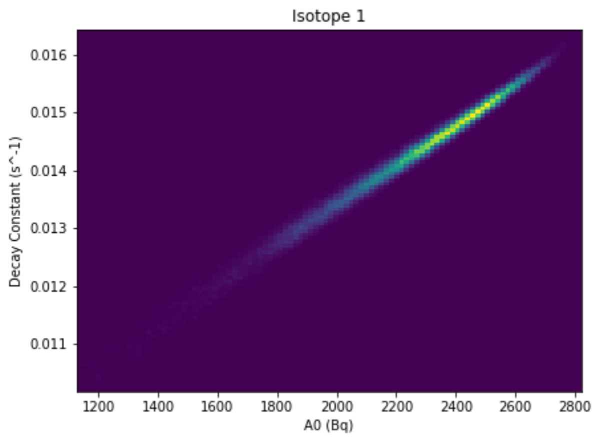
Graph 6: Activity at start for each MCMC iteration for the first isotope, for detector 1.

Once all the MCMC chains were checked, they were plotted in histograms so the median, upper and lower confidence intervals could be extracted for each parameter. See **Histogram 2** for and example of the chain in **Graph 6** plotted as a histogram.



Histogram 2: Frequency against Activity at start for detector 1, isotope 1 from the second dataset, with median, upper and lower Confidence Intervals

Before looking at the parameters output from the MCMC, there is one last check to be done. Correlation plots are created to see if there is any strange behaviour between parameters. The correlation between Decay Constant and A_0 can be seen in **Histogram 3** below. It shows a strong positive correlation between the 2 parameters for the first isotope.



Histogram 3: 2-Dimensional histogram showing the correlation between Decay Constant and A_0 for the first isotope, and first detector

We have now confirmed the MCMC is giving reliable results and do further analysis on the parameters. The values output from the MCMC are in **Table 12**.

	Decay Constant (s^{-1})	**Error (s^{-1})**	**A_0 (Bq)**	**Error (Bq)**
		Isotope 1		
Detector 1	0.01446	+0.00070 -0.00098	2313.1	+199.9 -290.0
Detector 2	0.01324	+0.00150 -0.00307	504.9	+104.6 -193.7
Detector 3	0.01667	+0.00062 -0.00104	222.1	+9.3 -18.6
Detector 4	0.01737	+0.00065 -0.00086	81.9	+2.8 -4.5
		Isotope 2		
Detector 1	0.04225	+0.00843 -0.00670	926.2	+279.2 -190.7
Detector 2	0.03735	+0.01343 -0.00927	302.4	+186.7 -99.6
Detector 3	0.14573	+0.18233 -0.07548	54.0	+15.6 -13.2
Detector 4	1.00484	+4.50406 -0.08348	20.1	+9.4 -9.2

Table 12: Best fit parameters from MCMC for a 2-isotope model, and associated errors.

In order to check if this 2-isotope model performs better than the 1-isotope, we need to look at the minimum Chi-Squared values and their associated AIC and BIC values. If these are lower than the 1 isotope model, we can confidently say the sample is made up of 2 sources. See **Table 13** for the AIC and BIC values

	Minimum χ^2	**AIC**	**BIC**
Detector 1	91.6	100.1	110.1
Detector 2	94.7	103.1	113.1
Detector 3	91.3	99.7	109.7
Detector 4	107.3	115.7	125.7

Table 13: Minimum Chi-Squared values and their associated AIC and BIC values for a double isotope model

The AIC and BIC values for the first 2 detectors show substantial reductions for a 2-isotope model compared to single isotope. However, the values for detectors 3 and 4 are very similar. This is due to the larger uncertainties in the data file for these detectors and these uncertainties can also be seen in the parameters for both single and double isotope models. For this reason, analysis will be focused on the first 2 detectors, and we can confidently say that the sample is made up of 2 isotopes.

The Half-Life associated with each isotope was then calculated to determine what the likely isotopes are which make up the sample. These values are found in **Table 14**. Using the first 2 detectors, with smallest errors, it can be deduced that the elements which make up the sample are ^{139}Eu and ^{168}W.

	Half-Life (s)	Error (s)
Isotope 1		
Detector 1	47.930	+10.533 -12.449
Detector 2	52.350	+17.628 -25.200
Detector 3	41.583	+8.012 -10.400
Detector 4	39.905	+7.737 -8.899
Isotope 2		
Detector 1	16.406	+7.328 -6.532
Detector 2	18.559	+11.127 -9.244
Detector 3	4.756	+5.320 -3.423
Detector 4	0.689	+1.460 -0.629

Table 14: Calculated Half-Life and uncertainties for each detector and isotope in the second dataset, from MCMC

The final part of the investigation involved determining whether the radiation in the second dataset was isotropic. This was calculated in the same way as for the first dataset, by performing a Weighted Least Squares and Bootsrap. The results were similar, thus proving the source for the second dataset is also isotropic. Powerlaw Index and Constants can be found in results **Table 15**.

	Value	Error
Powerlaw Index	-1.997	+0.008 -0.008
Powerlaw Constant	7.754	+0.153 -0.122

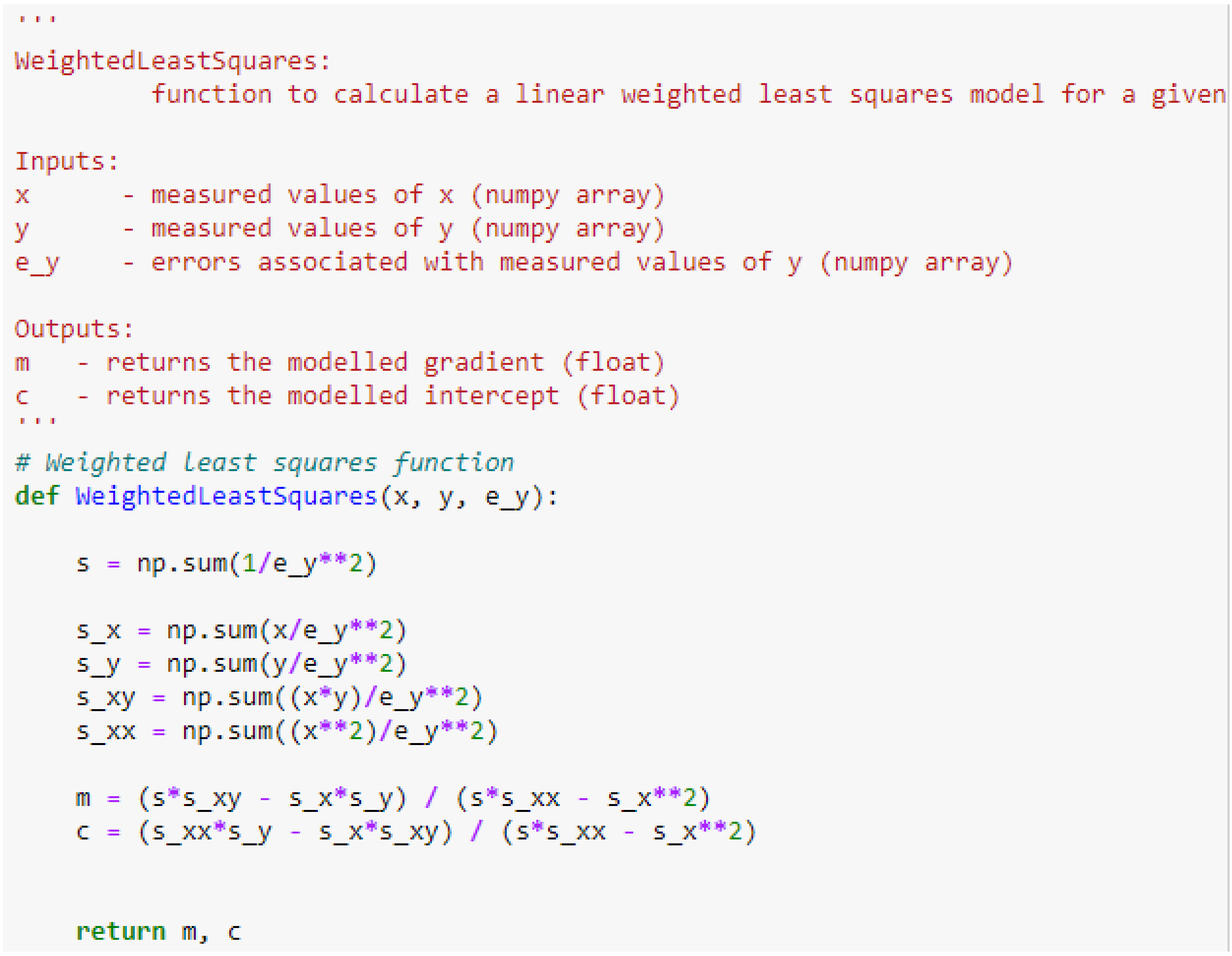
Table 15: Calculated Powerlaw Index and Constant for the second dataset

Finally, the number of particles in the 2-isotope sample was calculated. This was done in much the same way as for a single isotope however, the N_0 value was calculated separately for each isotope and combined to give a total value of 17,781,122 ± 143,167 particles.

4. Conclusions

By carrying out this investigation many useful methods of computational physics were used to accurately extract useful information from the dataset, along with its associated uncertainties. The isotopes making up the samples were deduced with a high degree of confidence and by using Information Criterions the best suited models were used for each scenario.

5. Appendix

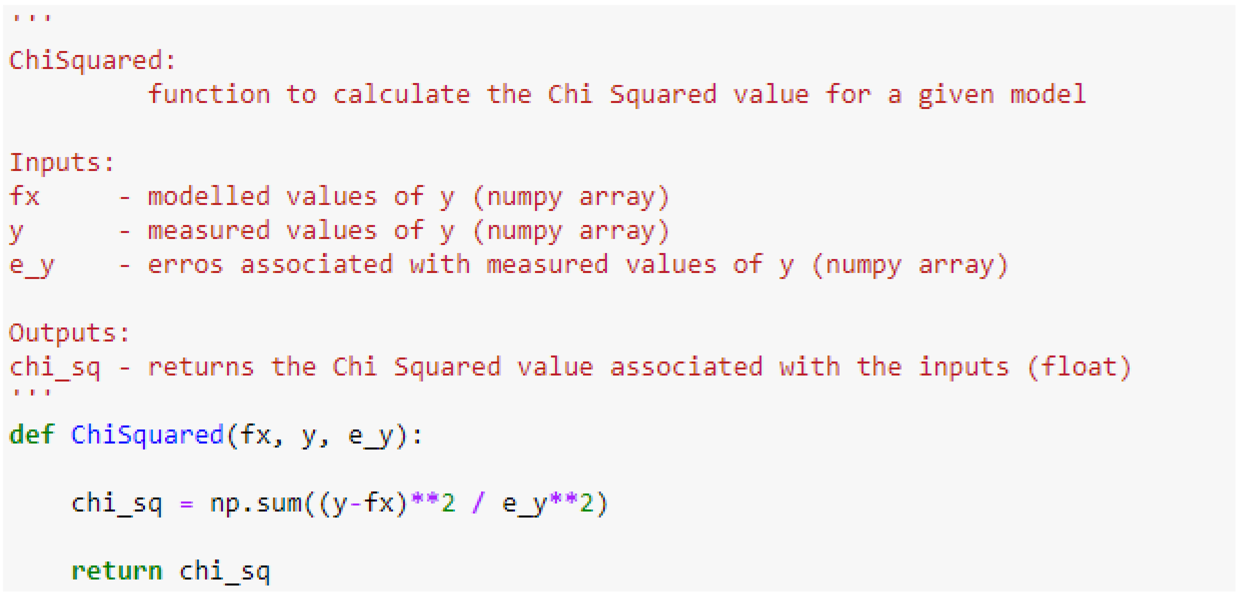


```python
'''
WeightedLeastSquares:
         function to calculate a linear weighted least squares model for a given

Inputs:
x       - measured values of x (numpy array)
y       - measured values of y (numpy array)
e_y     - errors associated with measured values of y (numpy array)

Outputs:
m    - returns the modelled gradient (float)
c    - returns the modelled intercept (float)
'''
# Weighted least squares function
def WeightedLeastSquares(x, y, e_y):

    s = np.sum(1/e_y**2)

    s_x = np.sum(x/e_y**2)
    s_y = np.sum(y/e_y**2)
    s_xy = np.sum((x*y)/e_y**2)
    s_xx = np.sum((x**2)/e_y**2)

    m = (s*s_xy - s_x*s_y) / (s*s_xx - s_x**2)
    c = (s_xx*s_y - s_x*s_xy) / (s*s_xx - s_x**2)


    return m, c
```

Appendix 1: Weighted Least Squares function for a linear model

```
'''
BootStrap:
         function to carry out a bootstrap analysis for a given model

Inputs:
residuals - residuals calculated for a given model (numpy array)
x       - measured values of x (numpy array)
m       - modelled gradient (float)
c       - modelled intercept (float)
e_y     - errors associated with measured values of y (numpy array)
iterations - number of iterations for bootstrap to run (int)

Outputs:
BS_m    - returns the gradient calculated at each bootstrap iteration (numpy array)
BS_c    - returns the intercept calculated at each bootstrap iteration (numpy array)
'''
# Bootsrap analysis function
def BootStrap(residuals, x, m, c, e_y, iterations):

    N = np.shape(residuals)[0]
    BS_m = np.zeros(iterations)
    BS_c = np.zeros (iterations)

    for j in range(0, iterations):
        temp_residuals = np.zeros(N)
        temp_y = np.zeros(N)
        temp_e_y = np.zeros(N)

        position = np.random.randint(0, N, N)

        for i in range(0, N):
            temp_residuals[i] = residuals[position[i]]
            temp_e_y[i] = e_y[position[i]]

        temp_y = m*x + c + temp_residuals
        BS_m[j], BS_c[j] = WeightedLeastSquares(x, temp_y, temp_e_y)

    return BS_m, BS_c
```

Appendix 2: Bootstrap analysis function for a linear model

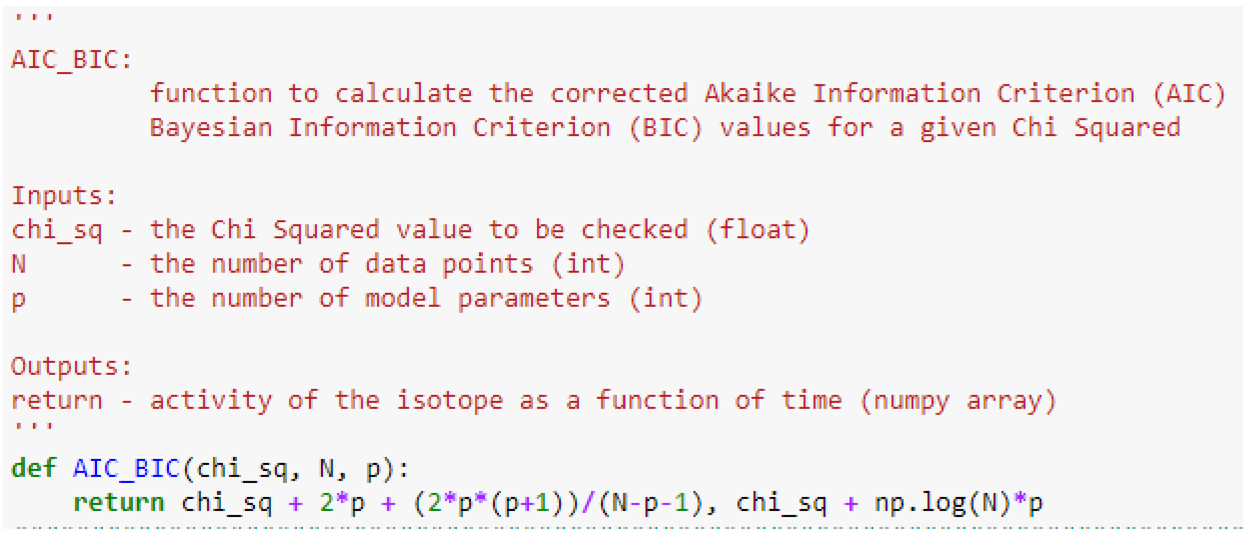


```python
'''
ChiSquared:
         function to calculate the Chi Squared value for a given model

Inputs:
fx      - modelled values of y (numpy array)
y       - measured values of y (numpy array)
e_y     - erros associated with measured values of y (numpy array)

Outputs:
chi_sq - returns the Chi Squared value associated with the inputs (float)
'''
def ChiSquared(fx, y, e_y):

    chi_sq = np.sum((y-fx)**2 / e_y**2)

    return chi_sq
```

Appendix 3: Function to calculate Chi-Squared statistic for a given model

```
'''
mcmc:    function to perform a simple MCMC using the Metropolis-Hastings Algorithm

Inputs:
func    - the function to be used for fitting. Function should accept 2 inputs
          the first is the x-values at which the model is calculated, the second
          is a numpy array for the parameters.
x       - the x-values at which the model should be calculated (numpy array)
y       - the value of the data for each x (numpy array)
sig     - the uncertainty on y (numpy array)
pars0   - The initial paramters from which to start the MCMC chain (numpy array)
stepsize - the stepsize for each of the parameters (numpy array)
nstep   - the number of steps in the MCMC chain

Outputs:
chain   - The MCMC chain as a (Nsteps, Npars) (numpy array)
'''
def mcmc(func,x,y,sig,pars0,stepsize,nstep=1e4):
    nstep=int(nstep)
    npars=pars0.shape[0]
    chain=np.zeros((int(nstep),npars))
    chi2=np.zeros(int(nstep))

    chain[0,:]=pars0.copy()
    mdl=func(x,pars0)
    chi2[0]=np.sum( (y-mdl)**2/sig**2)
    njump=0
    for i in range(1,nstep):
        pars_new=np.random.normal(chain[i-1,:],stepsize)
        mdl=func(x,pars_new)
        chi2_new=np.sum( (y-mdl)**2/sig**2)
        if (chi2_new > chi2[i-1]):
            p0=np.exp(-(chi2_new-chi2[i-1])/2.)
            p=np.random.uniform(0,1,1)
            if  p<=p0:
                chain[i,:]=pars_new
                chi2[i]=chi2_new
                njump=njump+1
            else:
                chain[i,:]=chain[i-1,:]
                chi2[i]=chi2[i-1]
        else:
            chain[i,:]=pars_new
            njump=njump+1
            chi2[i]=chi2_new
    print('Jump fraction: %.3f' %(njump/nstep))
    return chain
```

Appendix 4: MCMC function for any parameters of a given model



```
'''
AIC_BIC:
         function to calculate the corrected Akaike Information Criterion (AIC)
         Bayesian Information Criterion (BIC) values for a given Chi Squared

Inputs:
chi_sq - the Chi Squared value to be checked (float)
N      - the number of data points (int)
p      - the number of model parameters (int)

Outputs:
return - activity of the isotope as a function of time (numpy array)
'''
def AIC_BIC(chi_sq, N, p):
    return chi_sq + 2*p + (2*p*(p+1))/(N-p-1), chi_sq + np.log(N)*p
```

Appendix 5: Function to calculate AIC and BIC values