A Model For Experimentally Obtained Laser Data

Joseph Lyle Pritchard

University of Adelaide, Adelaide, Australia

Introduction

Understanding of laser behaviour has countless uses in a diverse array of areas and hence being able to model laser behaviour mathematically serves most useful. Investigation following aims to model experimentally obtained laser data using a system of 4 coupled differential equations.

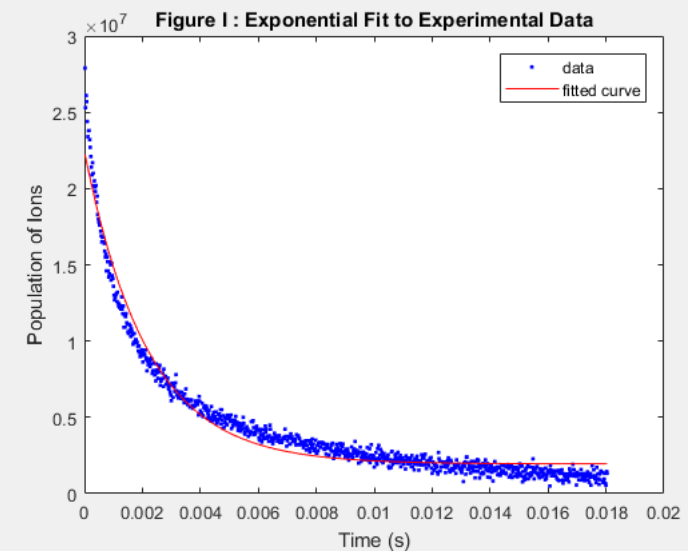
Procedure

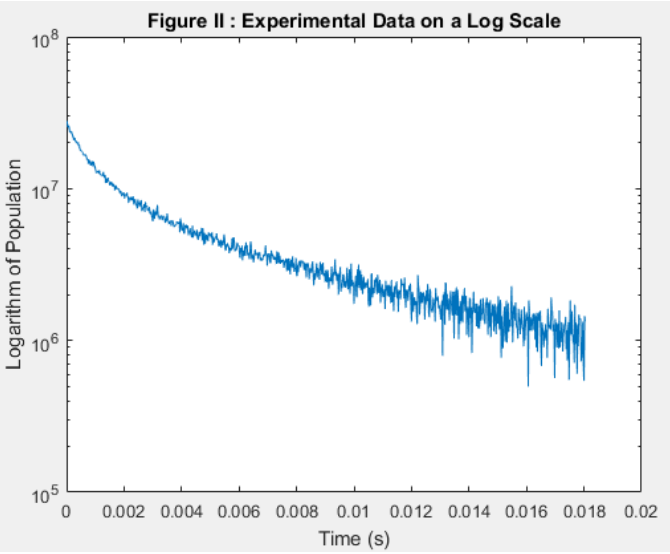
Note that for all produced results the MATLAB code used in the result creation can be viewed at the end of this report.

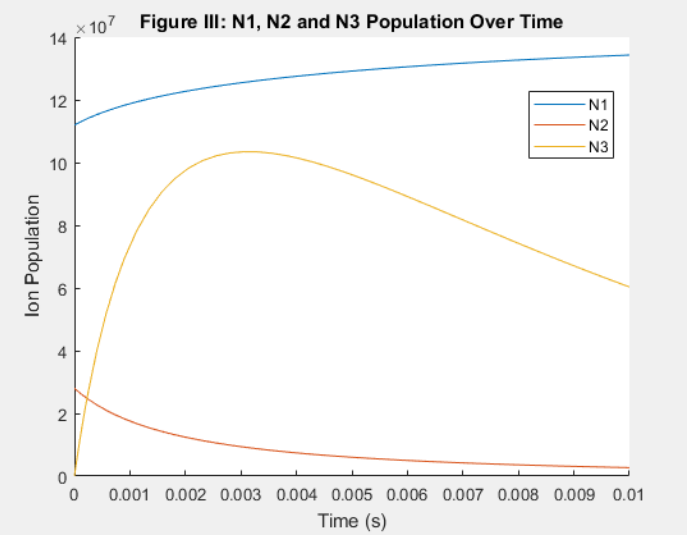
1. Project data is downloaded and saved in the working directory.
2. See Figure I and MATLAB Script attached.
3. In plotting N2 against time adopting N2 to be plotted on a log scale we note close linear correlation following the time marking of 200 units. Before this value we note a curved relation, perhaps fitted by an exponential. As such it is reasonable to assume low values are fitted by a steep decay while higher time values are fitted by a base 10 logarithmic function. These results can be viewed in Figure II.
4. A model will be constructed which follows the Physics specified.
5. We know that the transfer of ions from band 4 into band 3 is much larger than the transfer from band 4 to anywhere else. As such it is reasonable to assume that once an ion enters band 4 it is immediately transferred into band 3. This is modelled by treating bands 3 and 4 as a single band and further removing all terms corresponding to the transfer of ions from band 4 elsewhere.
6. Using the initial conditions specified in the outline the constructed model is solved using MATLAB functionality to produce the populations of bands N1, N2 and N3 over time as can be seen in Figure III. Note for ease of visibility the population of N2 is scaled up by a factor of 20 in this figure.
7. Previously used function is embedded within a parent function capable of accepting two parameters $N2_0$ and C_{up} and a time vector returning the values of the N2 population for each of the time values given in the time vector. To ensure optima functionality the function is tested for varying values of the two parameters as well as the input time vector. The produced graph is shown in Figure IV.

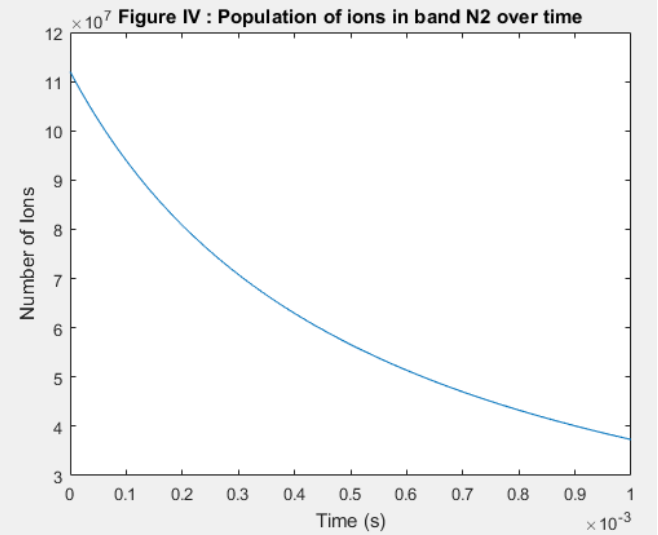
8. The mathematical model produced is now used to create a fit curve to the experimental data and test the models accuracy. As can be seen from the produced graph in figure V , the fit is closely matched to the experimental data suggesting the behaviour is accurately described by the physics sprouting the model.

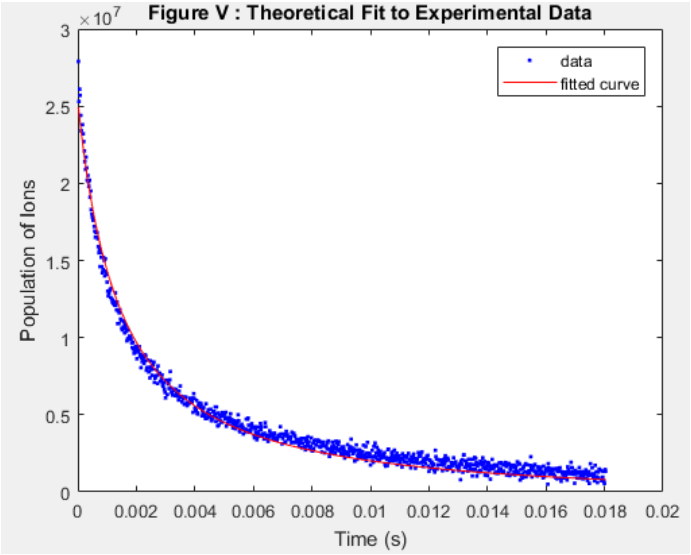
Graphs of experimental and theoretical data











MATLAB Code for Question 2

```
%Question 2
clear all;

%Import data
load('data.mat');
Time = Data.Time;
N2 = Data.N2;

%Define curve to be fitted to data
%A = Yaxis intercept, B = Decay rate, C = Asymptote
ExpFit = fittype('A*exp(-B*x) + C','coeff',{'A', 'B', 'C'});

%Define fit variables
FitVariables = [7e7, 9, 0];

%Fit to data and plot
[efit, gofexp] = fit(Time, N2, ExpFit, 'StartPoint', FitVariables, 'Lower', 0,
'Upper', 9e7);
plot(efit, Time, N2)
title('Figure I : Exponential Fit to Experimental Data')
xlabel('Time (s)')
ylabel('Population of Ions')
```

MATLAB Code for Question 3

```
%Question 3
clear all;
load('data.mat');
Time = Data.Time;
N2 = Data.N2;

%Define curve to be fit to data
%A = Yaxis intercept, B = Decay rate, C = Asymptote
ExpFit = fittype('A*exp(-B*x) + C','coeff',{'A', 'B', 'C'});

%Define fit variables
FitVariables = [7e7, 9, 0];

%Fit to data
[efit, gofexp] = fit(Time, N2, ExpFit, 'StartPoint', FitVariables, 'Lower', 0,
'Upper', 9e7);

%Attempt plotting on a Log scale
semilogy(Time,N2)
title('Figure II : Experimental Data on a Log Scale')
xlabel('Time (s)')
ylabel('Logarithm of Population')
```

MATLAB Code for Question 6

```
%Question 6
clear all
close all

%%%  Plot System of ODEs given in System_Des_6

%runtime
TFinal = 1e-2; %Seconds

%Total Ion poulation
NConc = 1.4e8; %Ions

%ICs
N0(1) = 0.8 * NConc;
N0(2) = 0.2 * NConc;
N0(3) = 0.0;
N0 = transpose(N0);

%Specify runtime
TSpan = transpose([0, TFinal]);

%Use ode45 to solve system with initial
%conditions N0 and time span TSpan
[TOUT,YOUT] = ode45(@System_DEs_Q6, TSpan, N0)

%Scale N3 to make behaviour visible
YOUT(:,3) = 20 * YOUT(:,3);

%Plot solution curves to ODE system
figure;
hold on
LineN1 = plot(TOUT, YOUT(:,1));
Label1 = "N1";
LineN2 = plot(TOUT, YOUT(:,2));
Label2 = "N2";
LineN3 = plot(TOUT, YOUT(:,3));
Label3 = "N3";
legend([LineN1; LineN2; LineN3], [Label1; Label2; Label3]);
title("Figure III: N1, N2 and N3 Population Over Time");
xlabel("Time (s)");
ylabel("Ion Population");
hold off
```

External Function for Question 6

```
function Result = System_DEs(t, Y)
%Y a vector of initial conditions
%t a scalar starting time

    w21 = 117;
    w31 = 105;
    w32 = 22;
    w41 = 0;
    w42 = 0;
    w43 = 0;
    C_Up = 8e-6;

    R(1) = w21*Y(2) + (w31 + w41)*Y(3) + C_Up*(Y(2))^2;
    R(2) = -w21*Y(2) + w32*Y(3) + w42*Y(3) - 2*C_Up*(Y(2))^2;
    R(3) = w43*Y(3) - (w32 + w31)*Y(3) + C_Up*(Y(2))^2 - (w43 + w42 + w41)*Y(3);
    Result = transpose(R);
end
```

MATLAB Code for Question 7

```
clear all
close all

%%%  Plot Population of N2 over time  %%%

%runtime
TFinal = 1e-3; %Seconds

%Total Ion poulation
NConc = 1.4e8; %Ions

%Define parameter variables
N2_0 = 0.8;
C_UP = 8e-6;
T = linspace(0, TFinal, 100);

%Find number of ions in N2 over time
N2 = Find_N2(T, N2_0, C_UP);

%Plot solution curves to ODE system
plot(T, N2);
title("Figure IV : Population of ions in band N2 over time");
xlabel("Time (s)");
ylabel("Number of Ions");
```

External Functions for Question 7

```
function N2 = Find_N2(T, N2_0, C_Up)
    %runtime
    TFinal = 1; %Seconds

    %Total Ion population
    NConc = 1.4e8; %Ions

    %Create handle which passes C_Up as a constant
    Current_EQn = @(T, N2_0) System_DEs_Q7(T, N2_0, C_Up)

    %ICs
    N0(1) = (1 - N2_0) * NConc;
    N0(2) = N2_0 * NConc;
    N0(3) = 0.0;
    N0 = transpose(N0);

    %Specify runtime
    TSpan = transpose([0, TFinal]);

    %Output N2 for specified t values given by T
    N_All = transpose(deval(ode45(Current_EQn, TSpan, N0), T));
    N2 = N_All(:,2);
end
```

```
function Result = System_DEs(t, Y, C_Up)
%Y a vector of initial conditions
%t a scalar starting time
    w21 = 117;
    w31 = 105;
    w32 = 22;
    w41 = 0;
    w42 = 0;
    w43 = 0;

    %Equations
    R(1) = w21*Y(2) + (w31 + w41)*Y(3) + C_Up*(Y(2))^2;
    R(2) = -w21*Y(2) + w32*Y(3) + w42*Y(3) - 2*C_Up*(Y(2))^2;
    R(3) = w43*Y(3) - (w32 + w31)*Y(3) + C_Up*(Y(2))^2 - (w43 + w42 + w41)*Y(3);
    Result = transpose(R);
end
```

MATLAB Code for Question 8

```
clear all;
load('data.mat');
Time = Data.Time;
N2 = Data.N2;

%Define curve to be fit to data
Fit = fittype(@(N2_0, C_Up, x) Find_N2(x,N2_0, C_Up));

%Fit to data
[efit, gofexp] = fit(Time, N2, Fit,'Lower',[0,8e-7],'Upper',[10e8,8e-
5],'StartPoint',[10e3 8e-6]);
plot(efit, Time, N2)
title('Figure V : Theoretical Fit to Experimental Data')
xlabel('Time (s)')
ylabel('Population of Ions')
```