**MAT5OPT Optimisation**

*Authors:*

Michael Le

Ruohan Xu

Vignesshwar Venkatachalam

April 2024

**Abstract:**

COVID-19 is considered as one of the deadly diseases commonly known as “SARS-CoV-2 Virus. The first case occurred in “Wuhan, China” in 2019 which later spread across the world in early 2020 where all countries in the world are remained in lockdown to reduce the spread of the virus among many patients and victims. This study is compromised from our la Trobe University Professor Peter Can Der Kamp. Is to understand that our goal is apply the following mathematical models for Exponential growth and the Verhulst logistic growth using the least squares method for any one of the COVID-19 data from GitHub, Datahub and Europa using MATLAB. We wanted to examinate at least one of the countries to understand their patterns and trends occurring at specific times due to social distancing and superspreader events by “flattening the curve” at different time-intervals. This can be achieved by applying and comparing distinct types of optimisation methods that was covered in video tutorials and workshop material. Using the gradient or steepest descend method, Newton’s method, the downhill simplex method, or any similar optimisation methods if necessary to achieve this study for the investigation.

**Introduction:**

The purpose of the investigation is to use the least square method for any of the COVID-19 data according to the assessment documentation. Comparing the two distinct types of models by further analysis and comparison to use the proper optimisation method at various times. This can enable to make decision making to influence and enable many health professionals, biostatisticians, and medical researchers to reduce the number of COVID-19 cases from spreading across the world. In this case, were focusing on one country taking place within the United States of America.

**Methodology:**

Topics to do for the least squares method which was covered in Weeks 4 to 6 Material for MAT5OPT on LMS (Learning Management System). We first must import the csv files holding the number of COVID-19 Cases and dates having years from 2020 to 2023 data from Github under “[CSSEGISandData”](https://github.com/CSSEGISandData) from the GitHub repository from the” University at John Hopkins”. By using the approximate optimisation algorithms for all models using the least squares method onto the two mathematical models shown below in this figure below, as well for other models.

A white paper with black text

Description automatically generated

To find the total number of infected people at t=0, the whole dataset including all number of infected cases from year 2020 to year 2023 is applied to the exponential growth model. However, because the number of infected people is unknown, so the starting point is t=1 instead of t=0. In appendix D (add code we used for exponential growth model).

Assumed mathematical knowledge such as linear algebra is needed to solve the linearised exponential growth model by using least squares regression, we get the value of ln(p) is 15.26 and the value of g is 0.0037. Transforming back into the non-linearised exponential growth. Since we get p = exp(ln(p)) gives us p = 4257569, Thus giving us the full equation for the non-linearised *exponential growth model*.

Furthermore, for the *linear regression model* we get the following coefficients of -9817547 for the intercept and 113838 for the gradient gives us,

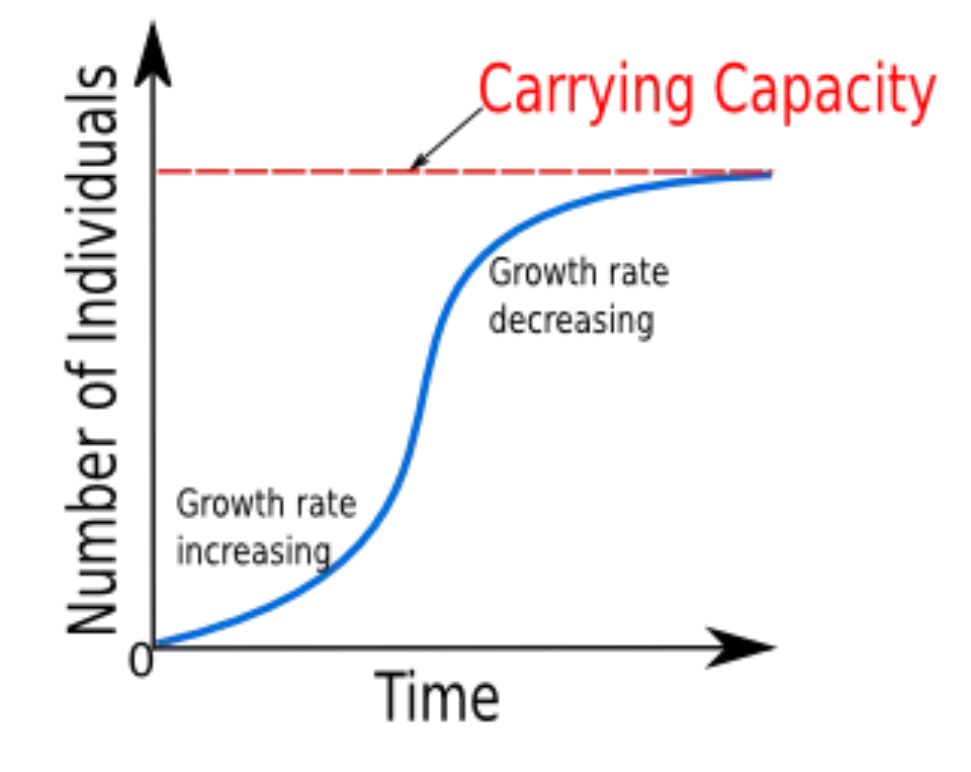
In addition, for the *polynomial fit using degrees of 4*, we get the following coefficients –3.1406e-04, 0.5176, -199.6474, 1.0128e+05 and -3.2042e+06 gives us, the full equation.

Lastly for the *Verhulst Model (logistic model)*, we take the limit when t approaches to infinity in finding c for the carrying capacity we use p and g coefficients from the exponential model gives us for the final model

**Implementations:**

As mentioned in the above, for the Verhulst logistic growth model, the p and g obtained from the exponential growth model are the same as in the Verhulst logistic model. As a result, the last parameter needs to be solved is c, which is the carrying capacity.

To solve for c, a function for Verhulst logistic model is created as shown in appendix E as well as the plot. An initial guess value is assigned to c, then the model is plotted to see if it fits the dataset. Then, lsqcurvefit function is used to perform curve fitting to estimate the real value of c as shown in appendix F. The estimated c value is 4.0767e+06 according to the result of the code in appendix F through trial and error. However, another approach to find c is we observe from asymptote the last record of cases from the full data occurred in this figure where the growth rate is slowing decreasing for the full dataset, we find out that c is 103802702 for the final dataset.



One student suggested to use the polynomial fit, which fits the data extremely well compared to other mathematical models (seen in Appendix D). Taken within timeframes from [0,225], [400,500] and [750,1060]. Next to the linear model is appropriate fit between intervals from [150,400], and [900,1000] despite there are noticeable outliners. Where there is only one exponential period are practical useful metric to demonstrate how effective lock-down measures are taken place within 225 days during 2020 after the first case occurred in the United States alone.

**Algorithms:**

Using the different optimisation algorithms, applying the least squares method for all the mathematical models we have decided to apply three algorithms the golden search method, downhill simplex method and Newton Method. After running the codes from Appendix I. For our findings for the downhill simplex method for the 2-simplex method for the full dataset. For the Exponential Growth is [874.126556, 0.011630]. Polynomial our is [113844.353241, -9820522.209922], Verhulst Growth (logistic) our [68697.151332, 0.012823], and lastly for the linear model, is [113844.353241, -9820522.209922]. Although this method does not require any derivatives, the purpose of the downhill simplex method is to achieve the optical vertex through Reflection, Expansion, Contraction and Shrinking for all the data-points from the dataset.

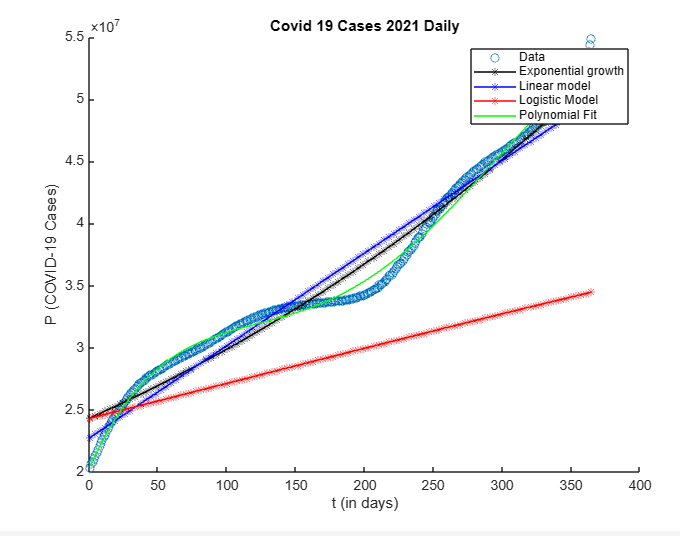
For the golden search method refer from Appendix H takes the number of the iterations until we find our desired minimiser, for the linear model it takes 42 iterations, the minimiser is 530.499998. For the polynomial model takes 42 iterations, the minimiser is approximately 557.585614. Exponential model After 42 iterations, the minimiser is approximately 661.161979. Lastly for the Verhulst Model takes 42 iterations, the minimiser is approximately 828.408593. Similarly for the downhill descent this algorithm does not take any derivatives from the objective function.

Furthermore, we have found the Minimiser to be around 692 for the Verhulst Logistic growth model from the adopted Newton Method being presented in Appendix J. As other models aren’t twice differentiable with respect to time, the Newton method couldn't be applied.

**Further Work/Suggestions:**

To further improve analysis, polynomial model is also used to compare with the exponential growth model. The code to generate polynomial model can be seen in appendix B, and original dataset, polynomial model, and exponential growth model are plotted together to compare in Figure 2. After plotting the data, the curve of polynomial model fits the observed data better than the exponential growth model. To further confirm this conclusion, sum of standard errors (SSE) can be computed for both models, which will quantify the discrepancies between the models and the data. Similarly, sum of squared error (SSE) can also be used to measure the discrepancies between the models and the data.

**Figure 2. COVID-19 US 2021 Dataset using the Exponential Growth, Polynomial Fit.**



There are also other aspects of the investigation can be done to enhance the analysis. The model can be modified to make forecast of COVID-19 infections. To be able to make forecast, time series data can be used to analyse trends, patterns, and relationships in the data over certain intervals of time (Franch-Pardo,2020) (i.e. taken yearly from 2020,2021,2022 and 2023) for further inspection. Where coefficients for p and g for the exponential growth model for example may change over time which is similar other mathematical models. Spatial analysis can also be used to explore the spread of COVID-19 in different countries other than the United States. Spatial analysis will help to provide insights into factors driving variations in different regions (Franch-Pardo,2020).

There could be other variables than the time alone including p and g as coefficients that could be considered to find the better approximated population at time zero i.e. p value. For instance, the variable measuring the population that showed the symptoms associated to COVID-19 but not actually tested positive for it could be taken into consideration.

**Conclusion:**

As a final remark, flattening the curve does not take in exponential phases for the entire dataset for the least squares method. We conclude that the best fit to flatten the curve is the polynomial fit with degrees of 4. As the number of cases in the US as the growth rate suddenly decreases stopping at 103802702 as of March 9th, 2023. By the value of the minimisers that we had derived from the Golden Search method for each model, the growth rate of the actual data points nearly evaluates to zero as well around the 500 to 600-day mark of the full dataset from COVID-19 Cases 2020 to 2023. We conclude that the best algorithm to optimise the sum of difference of residuals squared is Golden Search method as it optimises better than the other methods that we had adopted in terms of finding the minimiser of the objective function.

**References:**

*Coronavirus*. (2020, January 10). World Health Organisation (WHO). [https://www.who.int/healthtopics/coronavirus#tab=tab\_1](https://www.who.int/health-topics/coronavirus#tab=tab_1)

*COVID-19*. (2024, April 18). Yale Medicine. <https://www.yalemedicine.org/conditions/covid-19#:~:text=COVID-19%20is%20the%20first,anticipation%20of%20an%20influenza%20pandemic>

*Data.europa.eu*. (n.d.). The official portal for European data | data.europa.eu. <https://data.europa.eu/data/datasets/covid-19-coronavirus-data-weekly-from-17-december-2020%20?%20locale=en&locale=en>

*Datasets/covid-19: Novel coronavirus 2019 time series data on cases*. (n.d.). GitHub.

<https://github.com/datasets/covid-19>

Franch-Pardo, I., Napoletano, B. M., Rosete-Verges, F., & Billa, L. (2020). *Spatial analysis and GIS in the study of COVID-19. A review. Science of the total environment, 739, 140033.*

*Import data from file - MATLAB - MathWorks Australia*. (n.d.). MathWorks - Makers of MATLAB and Simulink - MATLAB & Simulink.

<https://au.mathworks.com/help/matlab/ref/importtool.html>

Johns Hopkins Coronavirus Resource Center. (2023, March 10). *CSSEGISandData/COVID-**19 : Novel coronavirus (COVID-19) cases, provided by JHU CSSE*. GitHub. <https://github.com/CSSEGISandData/COVID-19>

Knezevic, D. (n.d.). *Applied Mathematics 205 Unit I: Data Fitting*. <https://courses.seas.harvard.edu/courses/am205/fall13/AM205_unit_1_chapter_4.pdf>

*Novel coronavirus 2019*. (n.d.). Datahub. <https://datahub.io/core/covid-19>

La Trobe University: MAT5OPT Subject Notes. (2024). *Nonlinear regression: polynomials and linear combinations*. <https://dbb715huzqn8k.cloudfront.net/ac/68/ac68d7ae36cf1109b34c0e5e62d77232d50ca8ba?response-content-disposition=inline%3Bfilename%3D%22Week%204%20OPT%20notes.pdf%22&response-content-type=application%2Fpdf&Expires=1713623580&Signature=DH9xwv~NsU9zgh375rdcQZDHoLrAD-O3-vJIEOWraIO6DC5uBrKMt-L3FGueGBUTBgO~mPbvzHYrWzlYM5rBA67~j~Xd4ODRbE8oggxDS5RtEQAY3qgnGTR37I8ZU661BeQeVG~317WThL4nvONodfd8xrKg8TbyaBCwm0ZUiwQXGxquRVvnlEFU7QnsJAyhrypBSGDypG-o8C2-08ZWKi8WE9Ixo0KuDtxuFMRz9JkorTaq3n9b3UHAPjbaNu-Wrl5yLccS7sKDYSp~Z71rT4BIoGSDEp7MaP8XXDKYyCCcTLTVVQIEBThT0IZX8Gmhp54-TaTKoEFlWcpAIHi0QA__&Key-Pair-Id=APKAJMYMTS3PZ4IMRVZQ>

MathWorks. (2024). *Polynomial curve Fitting - MATLAB polyfit - MathWorks Australia*. MathWorks - Makers of MATLAB and Simulink - MATLAB & Simulink. <https://au.mathworks.com/help/matlab/ref/polyfit.html>

Xie, S. (2019, January 19). *Logistic growth model*. Medium. <https://medium.com/self-study-calculus/logistic-growth-model-96253b73ea37>

**Appendix:**

**Appendix A: COVID-19 US Cases from the 2021 Dataset**

% MAT5OPT Investigation

% Covering the COVID-19 cases in the US taking place in 2021

% Using the data from <https://github.com/CSSEGISandData>

%Were interested in

%t is time (in days) and

%P is the total number of COVID-19 Cases

%Step 1: Importing the data

dat = readtable('new.csv');

numberTable = dat(:,{'Time','Confirmed'});

t = numberTable(:,1);

P = numberTable(:,2);

%Convert them into arrays

t = table2array(t);

P = table2array(P);

%Step 2: Finding the coeffients for the following models,

%Finding coeffs for the polyfit model with degrees of 4

%NOTE: It will be the same as using the least squares method

%A = [t.^4 t.^3 t.^2 t ones(size(t))];

%coeffs = inv(A'\*A)\*A'\*P;

%Finding coeffs2 for the linearised exponential growth using least squares

A = [ones(size(t)) t];

logged = inv(A'\*A)\*A'\*log(P);

coeffs2 = [exp(logged(1)) logged(2)];

%Finding coeffs3 for the linear model using least squares

A = [ones(size(t)) t];

coeffs3 = inv(A'\*A)\*A'\*P;

%The carrying capacity, is the limit to infinity from P(t) where the

%number of cases approaches. Take the last row of the number of cases from

%the full dataset

c = 54907805;

figure

hold on

%Step 3:

%Plot the exponential model

scatter(t,P);

t=linspace(min(t), max(t));

plot(t, coeffs2(1)\*exp(t\*coeffs2(2)),'-k\*');

%Plot the linear model

plot(t,coeffs3(1)+t\*coeffs3(2),'-b\*');

%Plot the logistic model

plot(t,coeffs2(1)\*c./(coeffs2(1) + (c-coeffs2(1))\*exp(-coeffs2(2)\*t)),'-r\*');

xlabel('t (in days)');

ylabel('P (COVID-19 Cases)');

title('Covid 19 Cases 2021 Daily');

%Plot the polyfit model

yfit = polyval(coeffs,t);

plot(t,yfit,'g-');

legend({'Data','Exponential growth','Linear model','Logistic Model', ...

'Polynomial Fit'});

hold off

**Appendix A2: COVID-19 US Cases from the 2022 Dataset**

% MAT5OPT Investigation

% Covering the COVID-19 cases in the US taking place in 2022

% Using the data from <https://github.com/CSSEGISandData>

%Were interested in

%t is time (in days) and

%P is the total number of COVID-19 Cases

%Step 1: Importing the data

dat = readtable('data22.csv');

numberTable = table2array(dat);

t = numberTable(:,1);

P = numberTable(:,2);

%Step 2: Finding the coeffients for the following models,

%Finding coeffs for the polyfit model with degrees of 4

%NOTE: It will be the same as using the least squares method

A = [t.^4 t.^3 t.^2 t ones(size(t))];

coeffs = (A'\*A)\A'\*P;

%Finding coeffs2 for the linearised exponential growth using least squares

A = [ones(size(t)) t];

logged = inv(A'\*A)\*A'\*log(P);

coeffs2 = [exp(logged(1)) logged(2)];

%Finding coeffs3 for the linear model using least squares

A = [ones(size(t)) t];

coeffs3 = inv(A'\*A)\*A'\*P;

%The carrying capacity, is the limit to infinity from P(t) where the

%number of cases approaches. Take the last row of the number of cases from

%the full dataset

c = 100765362;

figure

hold on

%Step 3:

%Plot the actual data points

scatter(t,P);

%Plot the exponential model

t=linspace(min(t), max(t));

plot(t, coeffs2(1)\*exp(t\*coeffs2(2)),'-k\*');

%Plot the linear model

plot(t,coeffs3(1)+t\*coeffs3(2),'-b\*');

%Plot the logistic model

plot(t,coeffs2(1)\*c./(coeffs2(1) + (c-coeffs2(1))\*exp(-coeffs2(2)\*t)),'-r\*');

xlabel('t (in days)');

ylabel('P (COVID-19 Cases)');

title('Covid 19 Cases 2022 Daily');

%Plot the polyfit model

yfit = polyval(coeffs,t);

plot(t,yfit,'g-');

legend({'Data','Exponential growth','Linear model','Logistic Model', ...

'Polynomial Fit'});

hold off

**Appendix B: COVID-19 US Cases from the Full Dataset**

% MAT5OPT Investigation

% Covering the COVID-19 cases in the US taking place in

% from 2020 and 2023.

% Using the data from <https://github.com/CSSEGISandData>

%Were interested in

%t is time (in days) and

%p is the total number of COVID-19 Cases

%Step 1: Importing the data

dat = readtable('COVID\_19\_US\_Data\_2020-2023.csv');

numberTable = dat(:,{'Time','Cases'});

t = numberTable(:,1);

P = numberTable(:,2);

%Convert them into arrays

t = table2array(t);

P = table2array(P);

%Step 2: Finding the coeffients for the following models

%Finding coeffs for the polyfit model

%NOTE: It will be the same as

A = [t.^4 t.^3 t.^2 t ones(size(t))];

coeffs = inv(A'\*A)\*A'\*P;

%Alternatively, use the polyfit function

%coeffs = polyfit(t,P,4);

%Finding coeffs2 for the linearised exponential growth using least squares

A = [ones(size(t)) t];

logged = inv(A'\*A)\*A'\*log(P);

coeffs2 = [exp(logged(1)) logged(2)];

%Finding coeffs3 for the linear model using least squares

A = [ones(size(t)) t];

coeffs3 = inv(A'\*A)\*A'\*P;

%The carrying capacity, is the limit to infinity from P(t) where the

%number of cases approaches. Take the last row of the number of cases from

%the full dataset

c = 103802702;

figure

hold on

%Step 3:

%Plot the exponential model

scatter(t,P);

t=linspace(min(t), max(t));

plot(t, coeffs2(1)\*exp(t\*coeffs2(2)),'-k\*');

%Plot the linear model

plot(t,coeffs3(1)+t\*coeffs3(2),'-b\*');

%Plot the logistic model

plot(t,coeffs2(1)\*c./(coeffs2(1) + (c-coeffs2(1))\*exp(-coeffs2(2)\*t)),'-r\*');

xlabel('t (in days)');

ylabel('P (COVID-19 Cases)');

title('Covid 19 Cases 2020-2023 Daily');

%Plot the polyfit model

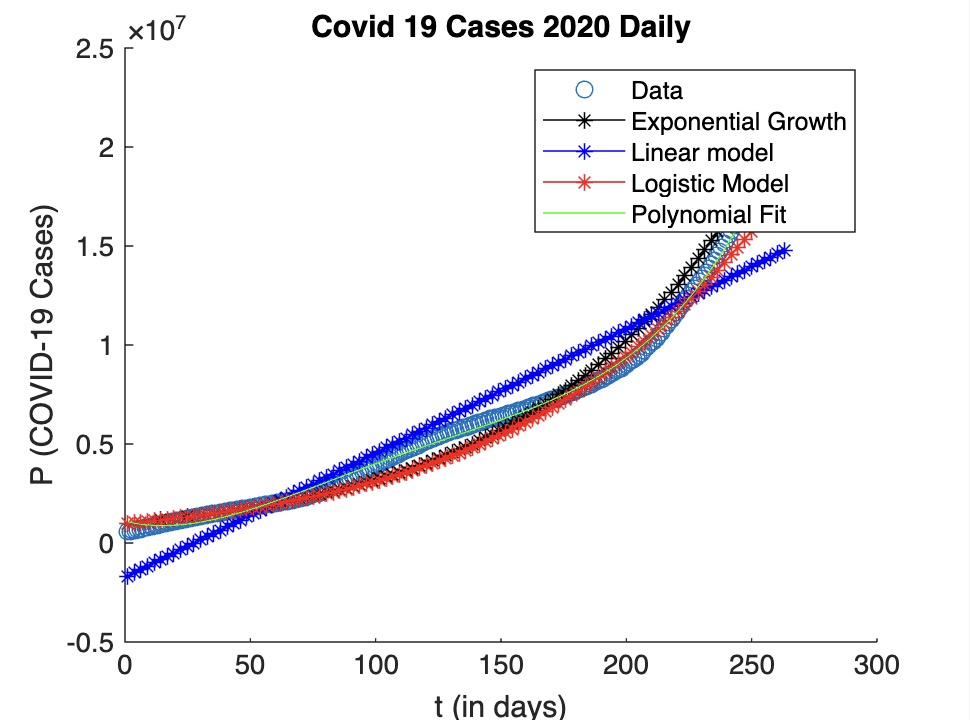
yfit = polyval(coeffs,t);

plot(t,yfit,'g-');

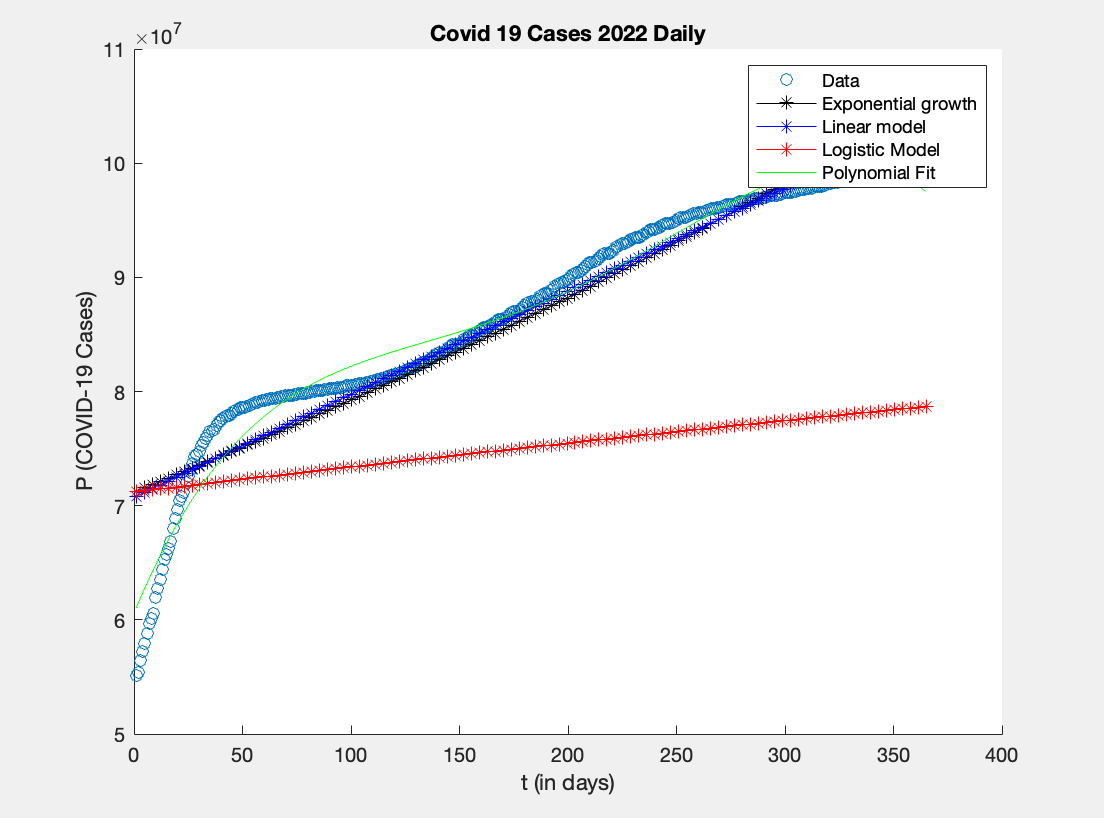
legend ({'Data','Exponential Growth','Linear model','Logistic Model','Polynomial Fit'});

hold off

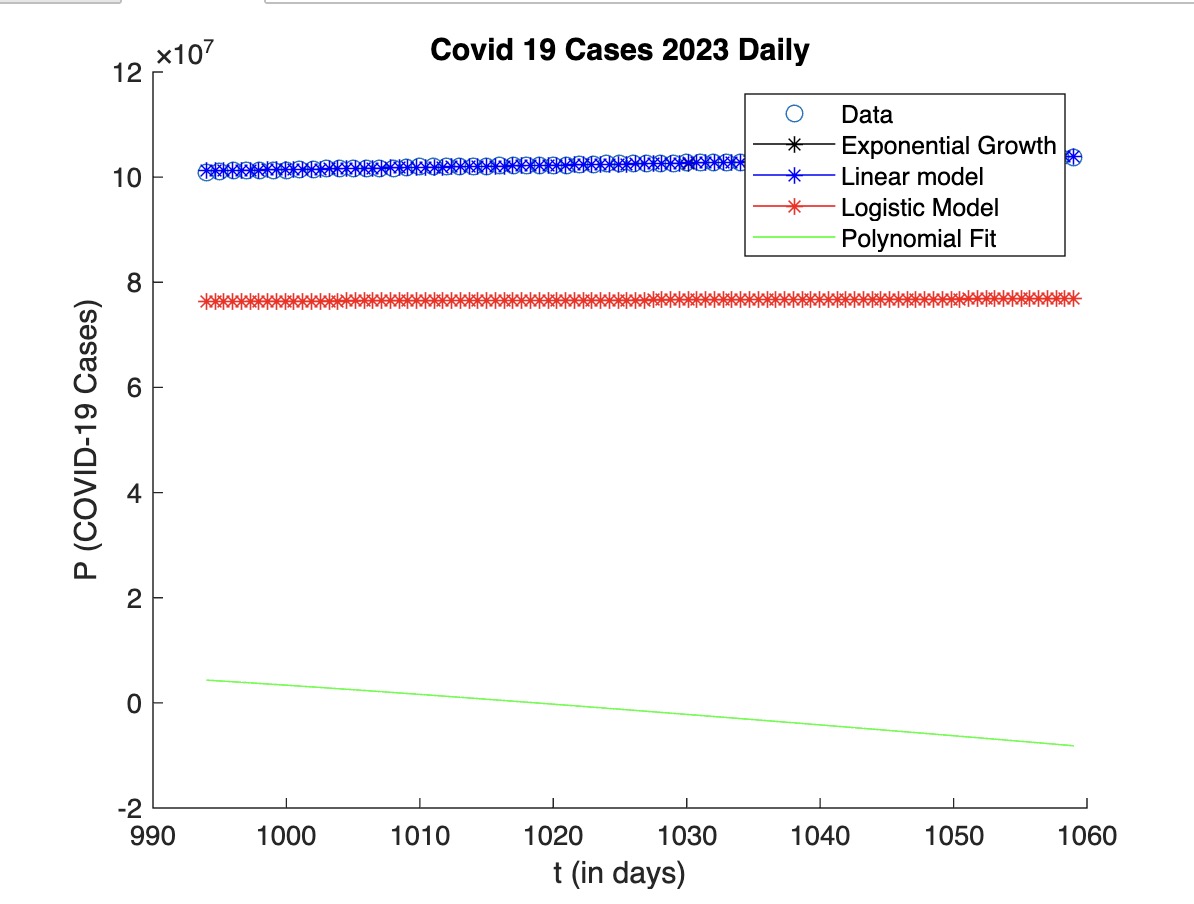
**1. Figure for the COVID-19 US 2020 Dataset using the Exponential Growth, Linear, Logistic and Polynomial Fit.**



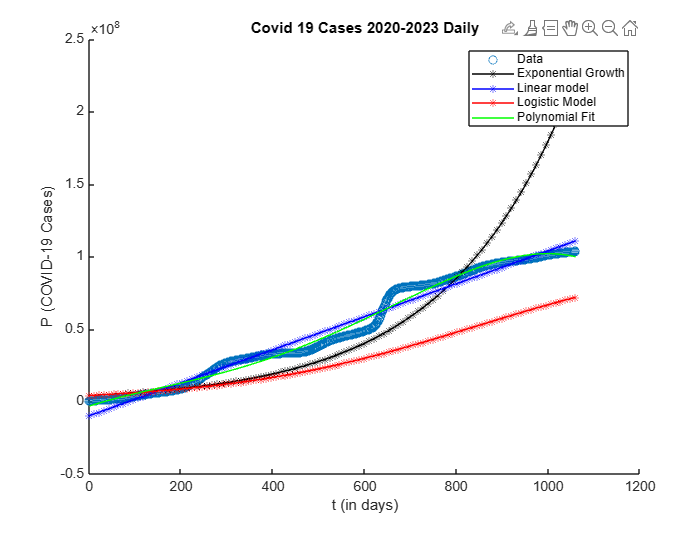
**2. Figure for the COVID-19 US 2022 Dataset using the Exponential Growth, Linear, Logistic and Polynomial Fit.**



**3. Figure for the COVID-19 US 2023 Dataset using the Exponential Growth, Linear, Logistic and Polynomial Fit.**



**Appendix D: Figure for the COVID-19 US Full Dataset using the Exponential Growth, Linear, Logistic and Polynomial Fit.**



**Appendix E:**

dat = readtable('COVID\_19\_US\_Data\_2020-2023.csv');

numberTable = dat(: {'Cases','Time'});

t = numberTable(:,2);

P = numberTable(:,1);

Z = log(P)

A = [ ones(size(t)]) t];

logged = inv (A.'\* A)\* A.'\* z;

a = exp (logged (1))

b = logged (2)

a =  
 4.0766e+06  
b =  
 0.0037

**Appendix F:**

function P\_t = verhulst\_logistic\_growth(t, p, g, c)

t = t (:);

% Compute P(t)

P\_t = p \* c./ (p + (c - p) \* exp (-g \* t));

end

% Define parameters

p = exp (15.26);

g = 0.0037;

c = 100; % Value of c, this is an initial guess.

% Define time points

t = 0:100;

% Compute P(t) using the Verhulst logistic growth model

P\_t = verhulst\_logistic\_growth(t, p, g, c);

% Plot the population growth over time

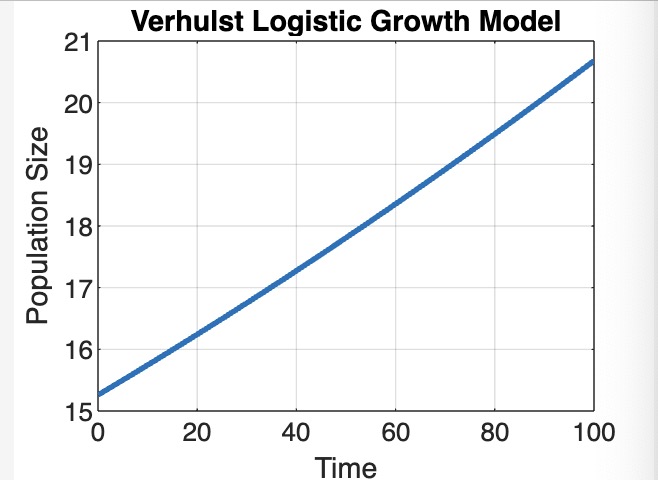
plot (t, P\_t);

xlabel('Time (t)');

ylabel('Population (P)');

title ('Verhulst Logistic Growth Model');

grid on;



**Appendix G:**

% Plot the population growth over time

t = numberTable(:,2);

P = numberTable(:,1);

% Define the Verhulst logistic growth model function

verhulst\_model = @ (params, t) params (1) \* params (3). / (params (1) + (params (3) - params (1)) \* exp(-params (2) \* t));

% Initial [p, g, c]

initial\_guess = [exp (15.2208), 0.037, 1000];

% Perform curve fitting using lsqcurvefit

estimated\_params = lsqcurvefit(verhulst\_model, initial\_guess, t, P);

% Extract the estimated carrying capacity (c)

estimated\_c = estimated\_params(3);

**Appendix H: Algorithm for the Golden Section Method Full dataset**

%We take the full dataset from 2020-2023

%Step 1: Importing the data

dat = readtable('COVID\_19\_US\_Data\_2020-2023.csv');

numberTable = dat(:,{'Time','Cases'});

t = numberTable(:,1);

P = numberTable(:,2);

%Convert them into arrays

t = table2array(t);

P = table2array(P);

%Step 2: Finding the coeffients for the following models

%Finding coeffs for the polyfit model

%NOTE: It will be the same as

%A = [t.^4 t.^3 t.^2 t ones(size(t))];

%coeffs = inv(A'\*A)\*A'\*P;

coeffs = polyfit(t,P,4);

%Finding coeffs2 for the linearised exponential growth using least squares

A = [ones(size(t)) t];

logged = inv(A'\*A)\*A'\*log(P);

coeffs2 = [exp(logged(1)) logged(2)];

%Finding coeffs3 for the linear model using least squares

A = [ones(size(t)) t];

coeffs3 = inv(A'\*A)\*A'\*P;

%The carrying capacity, is the limit from the Population where the

%number of cases approaches. Take the last row of the number of cases from

%the full dataset

c = 103802702;

%Step 3: Create the objective functions

%NOTE: if you want your test out desired objective function for f,

%uncomment that f, and leave out/the comments for the other mathematical models.

%EXPONENTIAL GROWTH:

%f = @(t) sum((P- (coeffs2(1)\*exp(t\*coeffs2(2)))).^2);

%LINEAR MODEL:

%f = @(t) sum((P - (coeffs3(1)+t\*coeffs3(2))).^2);

%VERHULST LOGISTIC MODEL:

f = @(t) sum((P - (coeffs2(1)\*c./(coeffs2(1) + (c-coeffs2(1))\*exp(-coeffs2(2)\*t)))).^2);

%POLYNOMIAL FIT DEGREE 4:

%f = @(t) sum((P - (polyval(coeffs,t))).^2);

%Step 4: Start the golden section algorithm

%INTIAL CONDITIONS:

%rho is fixed.

rho = (3- sqrt(5))/2;

%Take any

%Run initial values of a and d

%within the interval t = [a,d].

a = 0;

d = 1060;

iterations = 0;

while abs(a-d) > 2e-6

iterations = iterations+1;

b = a + rho\*(d-a);

c = d- rho\*(d-a);

if f(b) < f(c)

d = c;

else

a = b;

end

end

% The minimiser is in the interval [a,d], so we take the midpoint.

fprintf('After %d iterations, ', iterations)

fprintf('the minimiser is approximately %.6f.\n', (a+d)/2);

**Appendix I: Algorithm for the Downhill Deepest Descent Full dataset**

%We take the full dataset from 2020-2023

%Step 1: Importing the data

dat = readtable('COVID\_19\_US\_Data\_2020-2023.csv');

numberTable = dat(:,{'Time','Cases'});

t = numberTable(:,1);

P = numberTable(:,2);

%Convert them into arrays

t = table2array(t);

P = table2array(P);

%The carrying capacity, is the limit from the Population where the

%number of cases approaches. Take the last row of the number of cases from

%the full dataset

%Step 2: Create the objective functions

%NOTE: if you want your test out desired objective function for h,

%uncomment that h, and leave out/in comments

%for the other h's.

%EXPONENTIAL GROWTH:

h = @(coeffs) sum((P- (coeffs(1)\*exp(t\*coeffs(2)))).^2);

%LINEAR MODEL: (Works)

%h = @(coeffs3) sum((P - (coeffs3(1)+t\*coeffs3(2))).^2);

%VERHULST LOGISTIC MODEL:

c = 103802702;

%h = @(coeffs2) sum((P - (coeffs2(1)\*c./(coeffs2(1) + (c -coeffs2(1))\*exp(-coeffs2(2)\*t)))).^2);

%POLYNOMIAL FIT DEGREE 4:

%h = @(coeffs) sum((P - (polyval(coeffs,t))).^2);

%Step 3: Compute Downhill Simplex Method

%Initial conditions,

%NOTE for x1,x2,x3 dimensions must change according the

%number of unknown coefficents for each of the models.

%Refer to the steps 3.6. On the downhill simplex method.

x1 = [0; 0];

x2 = [0; 1];

x3 = [1; 0];

count = 0;

max\_iterations = 100;

alpha = 1;

gamma = 2;

rho = 1/2;

sigma = 1/2;

% Perform the downhill simplex method

for k = 1:max\_iterations

% Sort so that h(x1) < h(x2) < h(x3)

if h(x2) < h(x1)

temp = x1; x1 = x2; x2 = temp;

end

if h(x3) < h(x1)

temp = x1; x1 = x3; x3 = temp;

end

if h(x3) < h(x2)

temp = x2; x2 = x3; x3 = temp;

end

% Calculate centre of mass

xo = (x1 + x2)/2;

% Calculate reflection

xr = xo + alpha\*(xo- x3);

if h(x1) < h(xr) && h(xr) < h(x2)

x3 = xr;

elseif h(xr) < h(x1)

% Calculate expansion

xe = xo + gamma\*(xr- xo);

if h(xe) < h(xr)

x3 = xe;

else

x3 = xr;

end

else

if h(xr) < h(x3)

% Calculate outside contraction

xc = xo + rho\*(xr- xo);

else

% Calculate inside contraction

xc = xo + rho\*(x3- xo);

end

if h(xc) < h(x3)

x3 = xc;

else

% Shrink

x2 = x1 + sigma\*(x2- x1);

x3 = x1 + sigma\*(x3- x1);

end

end

end

fprintf('The minimiser is [%.6f, %.6f].\n', x1)

**Appendix J: Algorithm for the Newton Method Full dataset**

%Step 1: Importing the data

dat = readtable('2020to2023.csv');

numberTable = table2array(dat);

ti = numberTable(:,2);

P = numberTable(:,1);

%Step 2: Finding the coeffients for the following models,

%Finding coeffs for the polyfit model with degrees of 4

%NOTE: It will be the same as using the least squares method

A = [ti.^4 ti.^3 ti.^2 ti ones(size(ti))];

coeffs = (A'\*A)\A'\*P;

%Finding coeffs2 for the linearised exponential growth using least squares

A = [ones(size(ti)) ti];

logged = inv(A'\*A)\*A'\*log(P);

coeffs2 = [exp(logged(1)) logged(2)];

%Finding coeffs3 for the linear model using least squares

A = [ones(size(ti)) ti];

coeffs3 = inv(A'\*A)\*A'\*P;

%The carrying capacity, is the limit to infinity from P(t) where the

%number of cases approaches. Take the last row of the number of cases from

%the full dataset

c = 100765362;

figure

hold on

%Step 3:

%Plot the actual data points

%scatter(ti,P);

%Plot the exponential model

t=linspace(min(ti), max(ti));

%plot(ti, coeffs2(1)\*exp(ti\*coeffs2(2)),'-ro');

%Plot the linear model

%plot(ti,coeffs3(1)+ti\*coeffs3(2),'-y\*');

%Plot the logistic model

%plot(ti,coeffs2(1)\*c./(coeffs2(1) + (c-coeffs2(1))\*exp(-coeffs2(2)\*ti)),'-r\*');

xlabel('t (in days)');

ylabel('P (COVID-19 Cases)');

title('Covid 19 Cases 2020to2023');

%Plot the polyfit model

yfit = polyval(coeffs,ti);

%plot(ti,yfit,'g-');

%legend({'Data','Exponential growth','Linear model','Logistic Model', ...

%'Polynomial Fit'});

hold off

%newton1(@(ti)coeffs2(1)\*exp(ti\*coeffs2(2)),@(ti)coeffs2(2)\*coeffs2(1)\*exp(ti\*coeffs2(2)),1,747)

newton1(@(ti)coeffs3(1)+ti\*coeffs3(2),@(ti)coeffs2(2),1,42)

newton1(@(ti)coeffs2(1)\*c./(coeffs2(1) + (c-coeffs2(1))\*exp(-coeffs2(2)\*ti)),@(ti)-((c\*coeffs2(2)\*coeffs2(1))\*(coeffs2(1)-c)\*exp(coeffs2(2)\*ti))/(coeffs2(1)\*(exp(coeffs2(2)\*ti)-1)+c)^2,1,692)

function n = newton1(f,fd,n0,steps)

n = n0

for i =1:steps

n = n - f(n)/fd(n)

end

end