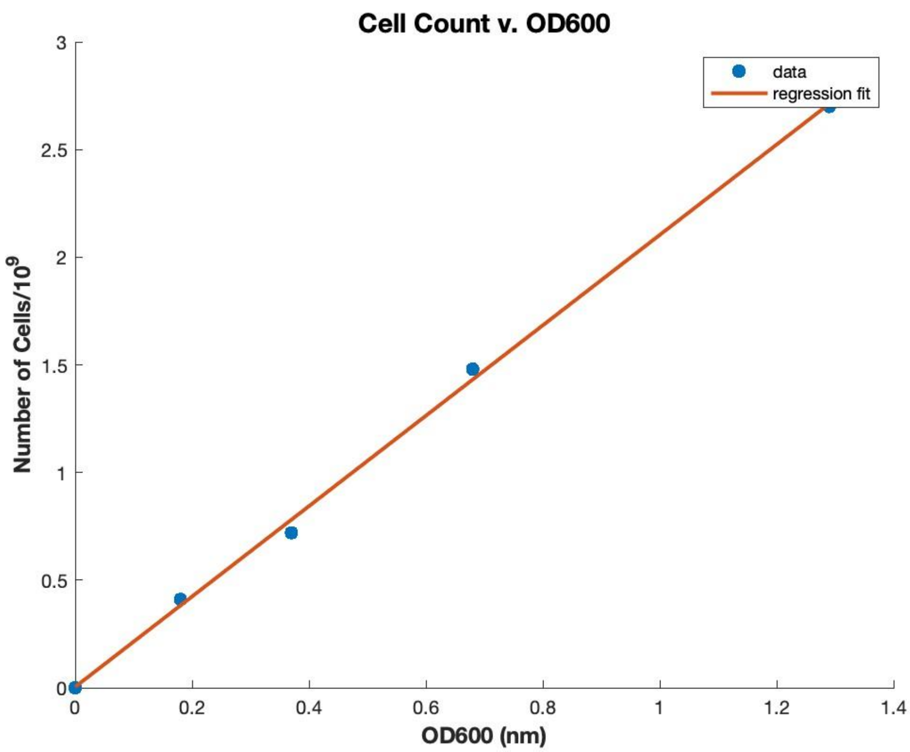
Robert Heeter
BIOE 446
1 September 2023

Lab 1: Numerical Methods and MATLAB Refresher

ANSWERS

1. **Conversion between optical density and cell mass data**
   [Cell count] = [OD600]*2.099364 + 0.003920
   C1 = 2.0994
   C2 = 0.0039

   Figure



2. **Logistic growth and parameter estimation**
   a. Counts = 0.2139, 0.2349, 0.4028, 0.7597, 1.2845, 1.9773, 2.7121, 3.3209, 3.9507, 4.1397, 4.1817

   b. g = 1.4089
      K = 4.2080

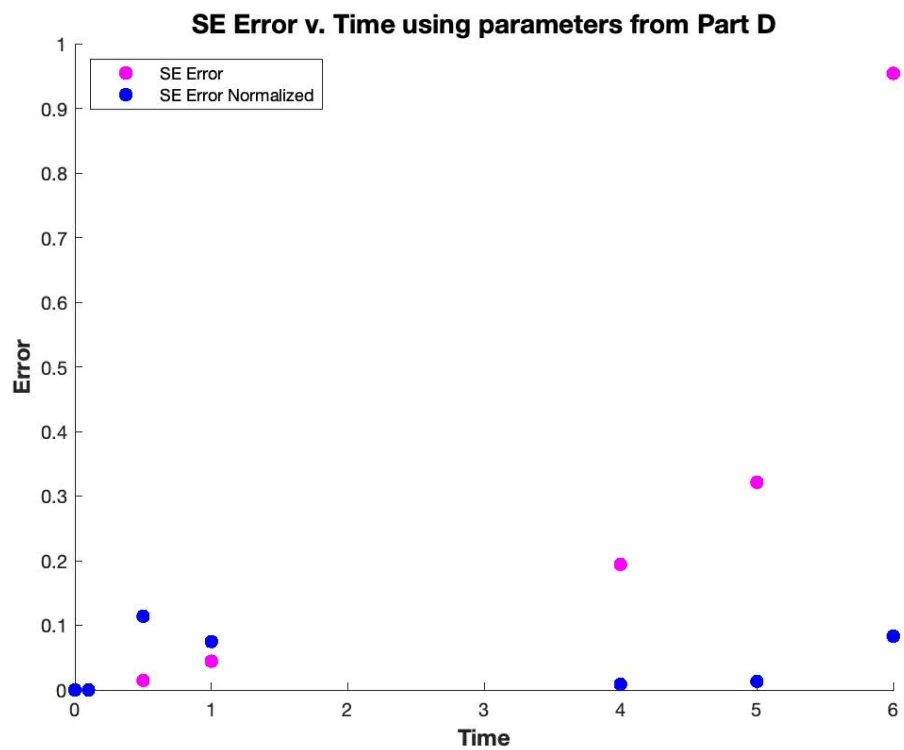
   c. g_5 = 1.4125
      K_5 = 4.0735
      The estimates of gamma and K are noticeably different from part B. Here g_5 = 1.4125 hours^-1 and K_5 = 4.0735 cells/10^9, whereas in part B, g = 1.4089 and K = 4.2080, respectively. When fitting logistic curves to data, it is important to have a good spread of population data over

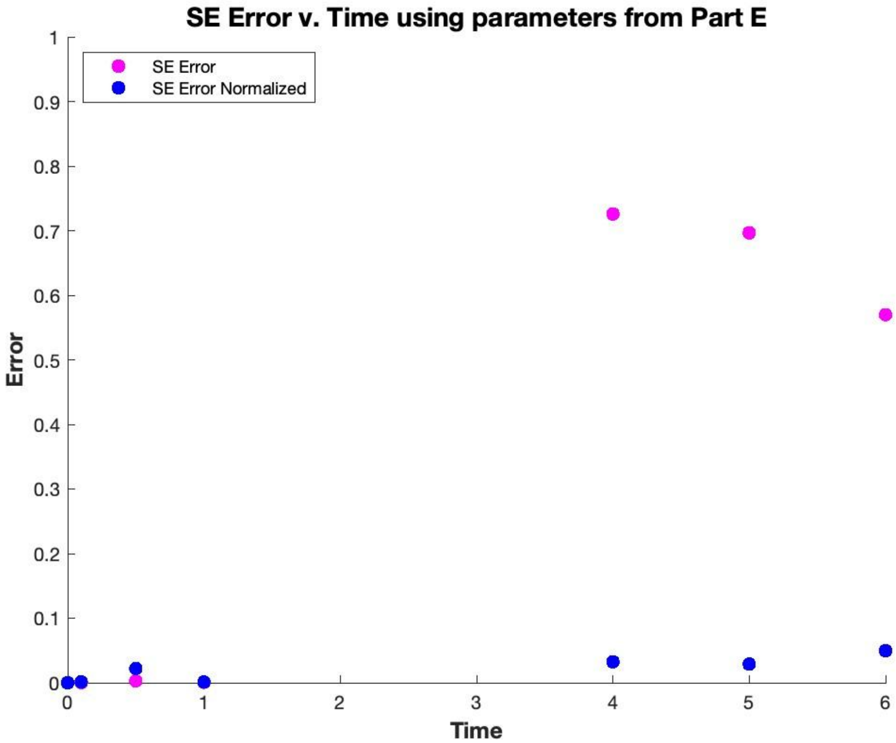
time, since it is difficult to estimate the final population carrying capacity using data from only a few initial time points. More data points improves the fit.

d. g_noisy = 1.6972
K_noisy = 4.3699
The estimates of gamma and K are noticeably different from part B. Here g_noisy = 1.6972 hours^-1 and K_noisy = 4.3699 cells/10^9, whereas in part B, g = 1.4089 and K = 4.2080, respectively. The noise in the data makes it very hard to accurately and confidently estimate the carrying capacity (K) and the growth rate (g), especially with so few data points (only 7 time points).

e. g_norm_noisy = 1.3626
K_norm_noisy = 4.1658
The estimates of gamma and K are noticeably different from part D. Here g_norm_noisy = 1.3626 hours^-1 and K_noisy = 4.1658 cells/10^9, whereas in part D, g_noisy = 1.6972 and K_noisy = 4.3699, respectively. This would imply that normalizing the error for fitting the data helps reduce the impact of noise on the calculated fit parameters.

f. Figures





The error for each data point using the normalized SE metric does not scale as the magnitude of the data increases, whereas the error for each data point using the non-normalized SE metric does (it grows, especially at 4, 5, and 6 hours). The normalized SE metric should be used in most cases since it better represents the relative size of the error when compared to the magnitude or "size" of each given data point.

CODE

```
close all
clear all
clc

%% Problem 1: Conversion between optical density and cell mass data
disp('PROBLEM 1')

OD600 = [0 0.18 0.37 0.68 1.29]; % nm
CFU = [0 0.41 0.72 1.48 2.7]; % cells/10^9

n = 1;
C = polyfit(OD600, CFU, n);
C1 = C(1)
C2 = C(2)
fprintf('[Cell Count] = [OD600]*%f + %f\n\n', C1, C2);

figure(1)
hold on
plot(OD600, CFU, '.', 'MarkerSize', 20)
plot(OD600, (OD600.*C(1)) + C(2), '-', 'LineWidth', 2)
legend('data', 'regression fit')
xlabel('OD600 (nm)', 'FontSize', 12, 'FontWeight', 'bold')
ylabel('Number of Cells/10^9', 'FontSize', 12, 'FontWeight', 'bold')
title('Cell Count v. OD600', 'FontSize', 14, 'FontWeight', 'bold')
hold off

%% Problem 2: Logistic growth and parameter estimation
time = [0 0.1 0.5 1 1.5 2 2.5 3 4 5 6]'; % hours
OD600 = [0.10 0.11 0.19 0.36 0.61 0.94 1.29 1.58 1.88 1.97 1.99]'; % nm

% Part A
disp('PROBLEM 2, PART A')

counts = polyval(C, OD600)

% Part B
disp('PROBLEM 2, PART B')

N = counts;
g_0 = 2; % hours^-1
K_0 = 4; % cells/10^-9

MSE_error_func = @(P) MSE_error(N, time, P);
[P_opt, ~] = fminsearch(@(P) MSE_error_func(P), [g_0, K_0]);

g = P_opt(1) % hours^-1
K = P_opt(2) % cells/10^-9

% Part C
disp('PROBLEM 2, PART C')
```

```
time_5 = [0 0.1 0.5 1 1.5]'; % first 5 data points
OD600_5 = [0.10 0.11 0.19 0.36 0.61]'; % first 5 data points

N_5 = polyval(C, OD600_5);

MSE_error_func = @(P) MSE_error(N_5, time_5, P);
[P_opt, ~] = fminsearch(@(P) MSE_error_func(P), [g_0, K_0]);

g_5 = P_opt(1) % hours^-1
K_5 = P_opt(2) % cells/10^-9

%{
The estimates of gamma and K are noticably different from part B. Here g_5
= 1.4125 hours^-1 and K_5 = 4.0735 cells/10^9, whereas in part B, g =
1.4089 and K = 4.2080, respectively. When fitting logistic curves to data,
it is important to have a good spread of population data over time, since
it is difficult to estimate the final population carrying capacity using
data from only a few initial time points. More data points improves the
fit.
%}

% Part D
disp('PROBLEM 2, PART D')

time_noisy = [0 0.1 0.5 1 4 5 6]'; % noisy data
N_noisy = [0.22 0.26 0.36 0.77 4.72 4.92 3.39]'; % noisy data

MSE_error_func = @(P) MSE_error(N_noisy, time_noisy, P);
[P_opt, f_noisy] = fminsearch(@(P) MSE_error_func(P), [g_0, K_0]);

g_noisy = P_opt(1) % hours^-1
K_noisy = P_opt(2) % cells/10^-9

[t, N_pred_noisy] = ode45(@(t, N) logistic(t, N, g_noisy, K_noisy), time_noisy,
N_noisy(1));

%{
The estimates of gamma and K are noticably different from part B. Here
g_noisy = 1.6972 hours^-1 and K_noisy = 4.3699 cells/10^9, whereas in part
B, g = 1.4089 and K = 4.2080, respectively. The noise in the data makes it
very hard to accurately and confidently estimate the carrying capacity (K)
and the growth rate (g), especially with so few data points (only 7 time
points).
%}

% Part E
disp('PROBLEM 2, PART E')

MSE_norm_error_func = @(P) MSE_norm_error(N_noisy, time_noisy, P);
[P_opt, f_norm_noisy] = fminsearch(@(P) MSE_norm_error_func(P), [g_0, K_0]);
```

```
g_norm_noisy = P_opt(1) % hours^-1
K_norm_noisy = P_opt(2) % cells/10^-9

[t, N_pred_norm_noisy] = ode45(@(t, N) logistic(t, N, g_norm_noisy, K_norm_noisy),
time_noisy, N_noisy(1));

%{
The estimates of gamma and K are noticably different from part D. Here
g_norm_noisy = 1.3626 hours^-1 and K_noisy = 4.1658 cells/10^9, whereas in
part D, g_noisy = 1.6972 and K_noisy = 4.3699, respectively. This would
imply that normalizing the error for fitting the data helps reduce the
impact of noise on the calculated fit parameters.
%}

% Part F
disp('PROBLEM 2, PART F')

SE_noisy_d = (N_noisy-N_pred_noisy).^2;
SE_noisy_norm_d = ((N_noisy-N_pred_noisy).^2)./((N_noisy).^2);

figure(2)
hold on
ylim([0, 1])
xlabel('Time', 'FontSize', 12, 'FontWeight', 'bold')
ylabel('Error', 'FontSize', 12, 'FontWeight', 'bold')
title('SE Error v. Time using parameters from Part D', 'FontSize', 14, 'FontWeight',
'bold')
set1 = plot(time_noisy', SE_noisy_d', 'm.', 'MarkerSize', 20);
set2 = plot(time_noisy', SE_noisy_norm_d', 'b.', 'MarkerSize', 20);
hold off

h = [set1(1) set2(1)]';
legend(h, 'SE Error', 'SE Error Normalized','location','northwest')

SE_noisy_e = (N_noisy-N_pred_norm_noisy).^2;
SE_noisy_norm_e = ((N_noisy-N_pred_norm_noisy).^2)./((N_noisy).^2);

figure(3)
hold on
ylim([0, 1])
xlabel('Time', 'FontSize', 12, 'FontWeight', 'bold')
ylabel('Error', 'FontSize', 12, 'FontWeight', 'bold')
title('SE Error v. Time using parameters from Part E', 'FontSize', 14, 'FontWeight',
'bold')
set1 = plot(time_noisy', SE_noisy_e', 'm.', 'MarkerSize', 20);
set2 = plot(time_noisy', SE_noisy_norm_e', 'b.', 'MarkerSize', 20);
hold off

h = [set1(1) set2(1)]';
legend(h, 'SE Error', 'SE Error Normalized','location','northwest')
```

```
%{
The error for each data point using the normalized SE metric does not scale
as the magnitude of the data increases, whereas the error for each data
point using the non-normalized SE metric does (it grows, especially at 4,
5, and 6 hours). The normalized SE metric should be used in most cases
since it better represents the relative size of the error when compared to
the magnitude or "size" of each given data point.
%}

%% Functions
function dN = logistic(t, N, g, K)
	dN = g.*N.*(1-N./K);
end

function MSE = MSE_error(N, time, P)
	g = P(1);
	K = P(2);
	N_0 = N(1);
	[t, N_pred] = ode45(@(t, N) logistic(t, N, g, K), time, N_0);
	MSE = sum((N_pred-N).^2);
end

function MSE_norm = MSE_norm_error(N, time, P)
	g = P(1);
	K = P(2);
	N_0 = N(1);
	[t, N_pred] = ode45(@(t, N) logistic(t, N, g, K), time, N_0);
	MSE_norm = sum(((N_pred-N).^2)./(N.^2));
end
```