Table of Contents

.... 1
Task 1: Integrating Data 1
Task 2: Integration Flexbility 1
Extra Stuff for Fun 3


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% MATH_151_Lab5
%--------------------------------------------------------------------------
% C Rocheleau, Colorado State University
% 9/26/2023
%--------------------------------------------------------------------------
% Answer key for MATH-151 Lab 6 for the Fall 2023 semester
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

close all; clear all; clc;
```

Task 1: Integrating Data

```
% Organize GPS data
gps_x = [0, 0.5, 1];
gps_y = [0, 0.3256, -0.1108];

len_x = max(gps_x) - min(gps_x);

% a) Approximate integral using Midpint rule
Midpt_int = gps_y(2)*len_x;

% b) Approximate integral using Trapezoid Rule
Trapz_int = (gps_y(1) + gps_y(end))*len_x / 2;

% c) Approximate integral using Simpson's rule
Simps_int = (gps_y(1) + 4*gps_y(2) + gps_y(end))*len_x / 6;

% d) Output our answers and guess what the real might be
fprintf('Midpoint \t Trapezoid \t Simpson''s \n %1.3f \t\t %1.3f \t \t %1.3f
 \r\n', ...
         Midpt_int, Trapz_int, Simps_int);

% I trust Simpson's Method the most, so I would think its near 0.2
```

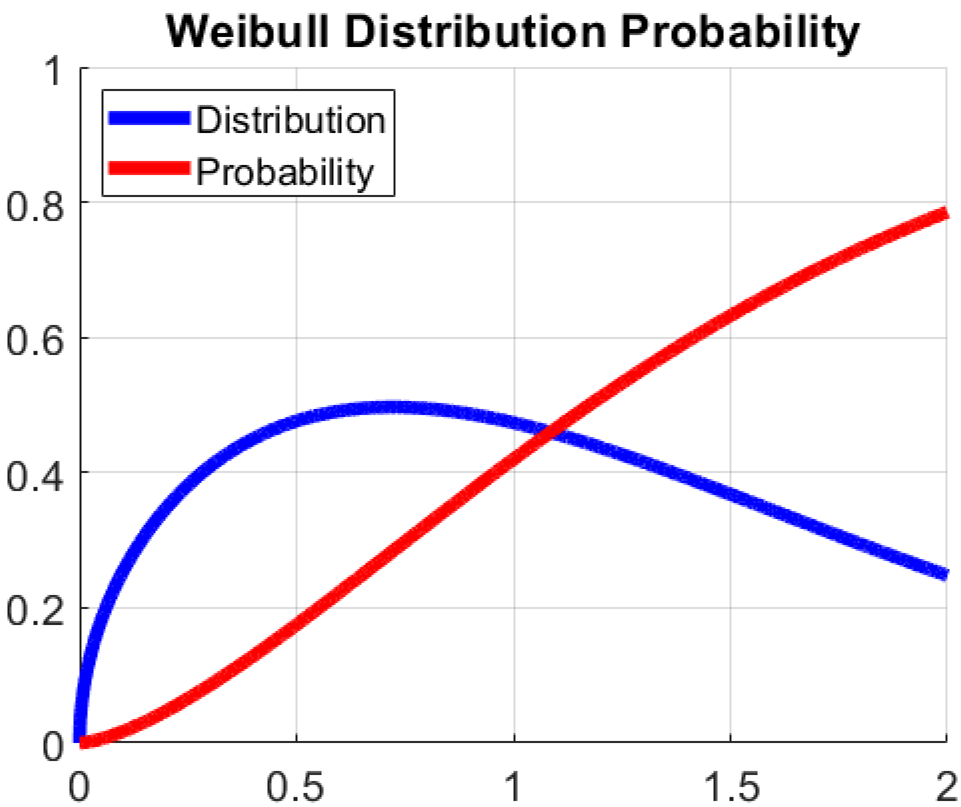
```
Midpoint   Trapezoid   Simpson's
 0.326    -0.055      0.199
```

Task 2: Integration Flexbility

```
% Set up an integration scheme for Weibull Distribtion between 0 and 2
```

```
k = 3/2; lambda = 3/2;       % Weibull Parameters
a = 0; b = 2;                % Integration Limits
N = 2001; h = (b-a)/N;      % Number and width of intervals for Riemann Sum
x = a:h:b;                   % Find edges of each of our intervals

% Initialize integral Sum (make vector so we can see antiderivative)
Weibull_int = 0*x;
fWeibull    = (k/lambda).*(x/lambda).^(k-1).*exp(-((x/lambda).^k));

% Loop through intervals computing Riemann sum
for iInt = 1:N
    % Grab Function values at left and right side of interval
    fL = fWeibull(iInt);
    fR = fWeibull(iInt + 1);

    % Update Using Trapezoidal rule
    Weibull_int(iInt+1) = Weibull_int(iInt) + (fL + fR)*h/2;
end

fprintf('The probablity of failing in 2 years is %1.6f \r
\n',Weibull_int(end));

% Plot the Weibull distributions Probability Density Function (PDF) and
% Cumulative Distribution Function (CDF)
figure(); hold on; grid on;
h_PDF = plot( x, fWeibull, 'b-','linewidth', 5);
h_CDF = plot( x, Weibull_int, 'r-','linewidth',5);
legend([h_PDF, h_CDF],
 {'Distribution', 'Probability'}, 'location','northwest');
title('Weibull Distribution Probability'); ylim([0 1]);
set(gca,'fontsize',16);
```

The probablity of failing in 2 years is 0.785528



Extra Stuff for Fun

Do it With Simpson's Rule

```
NSimps   = 1000; hSimps = 2/NSimps;      % We want 1000 divisions
xSimps   = linspace(0,2,2*NSimps+1);     % But that means we need 2001 points
 (1001 endpoints and 1000 midpoints)
fWeibull = (k/lambda).*(xSimps/lambda).^(k-1).*exp(-((xSimps/lambda).^k));

% We can be very fancy and actually do this as an inner product!
% Each endpoint will be a left and right side
SimpsWeights  = (2*hSimps/6)*ones(size(fWeibull));
% Midpoints of each division are even numbers
SimpsWeights(2:2:2*NSimps) = 4*hSimps/6;
% The endpoints only appear in one sum
SimpsWeights([1 end])      = hSimps/6;
% Inner Product takes care of multiplication and sum for us!
SimpsInt = SimpsWeights*fWeibull';

exact = 1 - exp(-(2/lambda)^k);
fprintf(['Using 2001 Points \n', ...
         'Trapezoidal Rule Error = %0.6f \n', ...
         'Simpson''s Rule Error = %0.6f \r\n'], ...
         exact - Weibull_int(end), exact - SimpsInt)
```

Using 2001 Points

```
Trapezoidal Rule Error = 0.000005
Simpson's Rule Error = 0.000002
```

Published with MATLAB® R2022b