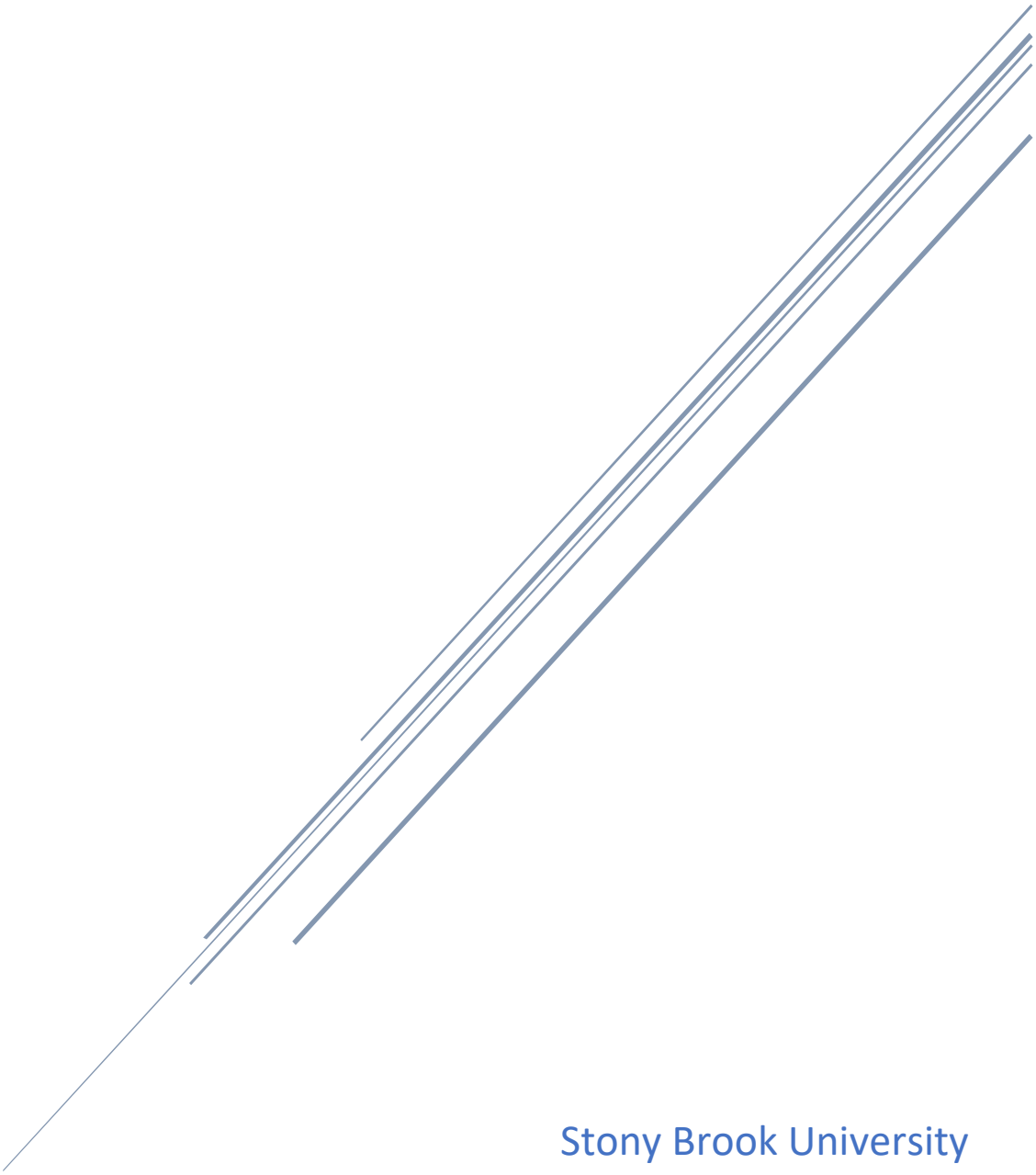
MEC 320 PROGRAMMING

ASSIGNMENT III

CHING WEI HUANG 114446084



Stony Brook University

MEC 320

Introduction:

Problem 24.30 on Page 702 of the textbook "Numerical Methods for Engineers." (Chapra and Canale, 2021) are what desired to solve for in this programming assignment. In the problem, a force distribution curve of a skyscraper during a storm is given.

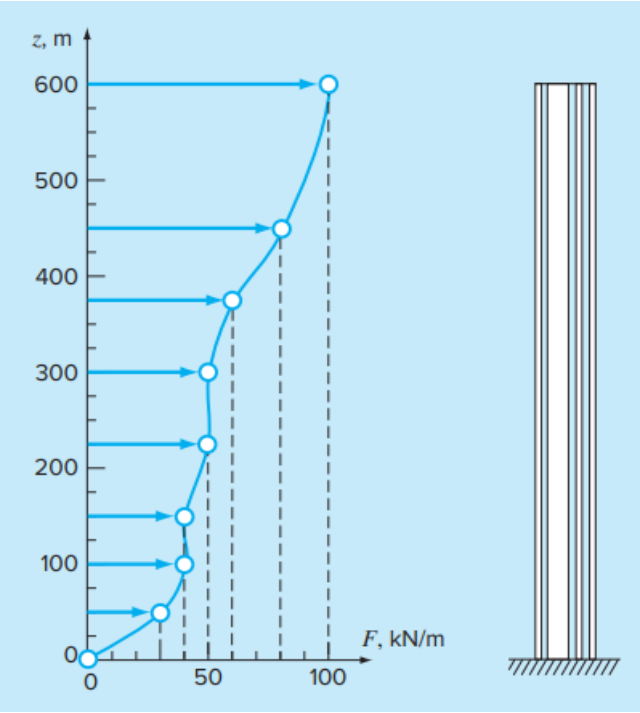


Figure 1. The force distribution curve of a skyscraper given by the problem (Chapra and Canale, 2021).

The problem asked to solve for the force and the line of force on the building. This assignment solves the problem using 2 numerical approaches, which are Trapezoidal Rule and Simpson 1/3. The calculation of both approaches is done by self-written MATLAB code. The results from each methods would then be compared in the discussion section.

Modeling:

The force distribution function over the building can be describe as

$$F \equiv F(z) \qquad (1)$$

where z is the height of the skyscraper in meter and F is the distributed force along the building in kN/m. There are 9 known measured data points on the function; Their values are shown in table 1.

	1	2	3	4	5	6	7	8	9
z	0	50	100	150	225	300	375	450	600

F(z)	0	30	40	40	50	50	60	80	100

Table 1. The known data points on the force distribution function.

The total force F_{net} acting on the building is

$$F_{net} = \int_{z=0}^{z=600} F(z)dz \quad (2)$$

Moreover, using the concept of resultant force, the system is equilibrium to the following graph.

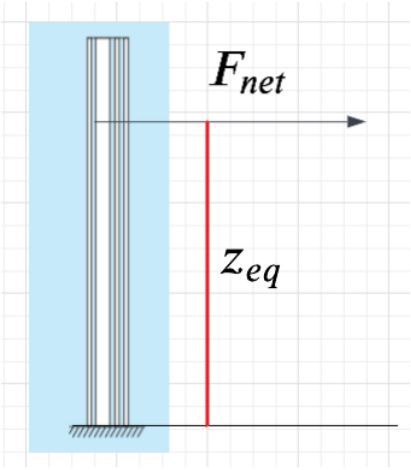


Figure 2. The force equilibrium system of the skyscraper, where F_{net} is the resultant force and z_{eq} is the location of the resultant force, which is also known as the line of force on the building

The F_{net} in figure 2 is the resultant force in *kN* and z_{eq} is the location of the resultant force in meter, which is also known as the line of force on the building. The line of force z_{eq} can be represent as

$$z_{eq} = \frac{M_{net}}{F_{net}} \quad (3)$$

M_{net} is the total moment contributed by the distributed forces in $kN \cdot m$, which can be calculated by

$$M_{net} = \int_{z=0}^{z=600} M(z)dz = \int_{z=0}^{z=600} z \cdot F(z)dz \quad (4)$$

Similar, calculated from table 1, There are 9 known data points on the function; Their values are shown in table 2.

	1	2	3	4	5	6	7	8	9
z	0	50	100	150	225	300	375	450	600
M(z)	0	1500	4000	6000	11250	15000	22500	36000	60000

Table 2. The known data points on the moment distribution function.

Problem Solving Algorithm:

First, it is clear to see from table 1 and 2 that the width *h* between each *z* data

points are not equal. This could be a problem while using Simpson 1/3. To overcome this problem, the Newton's interpolating polynomial method was chose to fit a function describing the force distribution curve function shown in figure 1. The detail MATLAB code for doing this is provided appendix A. The function got from this procedure is shown in figure 3.

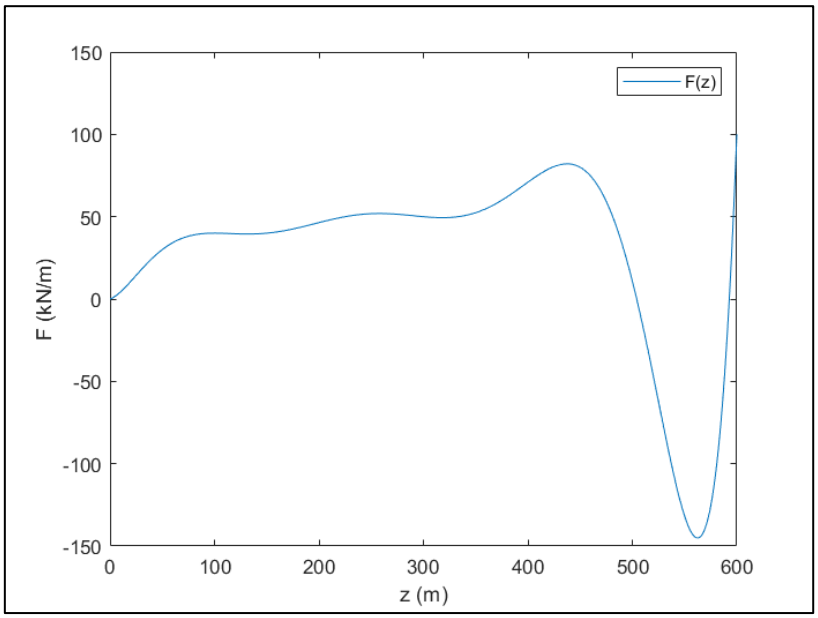


Figure 3. The force distribution curve function obtained by Newton's interpolating polynomial method over data points from table 1.

Compare this function with the curve from figure 1, the F value from z in interval $(450,600)$ are completely different. The integration from the function from fitting cannot represent the original problem. Another data point $(z, F(z))=(500,90)$ obtained by eye checking the curve from figure 1 was added into the fitting procedure. The result is as shown in figure 4.

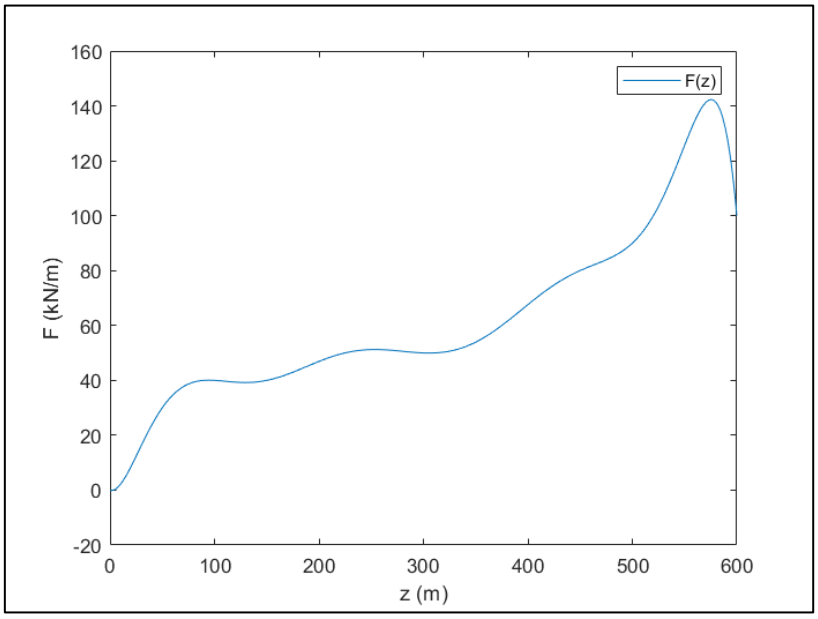


Figure 4. The force distribution curve function obtained by Newton's interpolating polynomial method over data points from table 1 and an addition point *(z, F(z))=(500,90)* from eye checking of the figure 1 curve.

Unfortunately, the *F* value from *z* in interval *(450,600)* are still largely different from the original curve. It is not a proper approach to solve the uneven width *h* issue using curve fitting.

To solve the uneven width *h* issue, data points are added on table 1 and 2 from the curve in figure 1 by eye checking. Values are taken each step of *25* on the *z* axis. That is, table 1 and 2 are extended and merged into table 3.

	1	2	3	4	5	6	7	8	9	10
z	0	25	50	75	100	125	150	175	200	225
F(z)	0	15	30	38	40	40	40	43	47	50
M(z)	0	375	1500	2850	4000	5000	6000	7525	9400	11250

	11	12	13	14	15	16	17	18
z	250	275	300	325	350	375	400	425
F(z)	50	50	50	52	57	60	70	75
M(z)	12500	13750	15000	16900	19950	22500	28000	31875

	19	20	21	22	23	24	25
z	450	475	500	525	550	575	600
F(z)	80	88	90	92	96	99	100
M(z)	36000	41800	45000	48300	52800	56925	60000

Table 3. The discrete data points used to solve the problem

Notice from table 3 that the width *h* between each *z* is now constant and equal to *25*, which means that the number of panels (segments) for both Trapezoidal Rule and Simpson 1/3 are *24* panels.

The MATLAB code for computing the Trapezoidal Rule and Simpson 1/3 are similar; Both these functions take in *2* arrays with size $1 \times n$ and return a number which is the numerical integral value. There exit a “minimum unit” for both methods, which are 1 and 2 panels for Trapezoidal Rule and Simpson 1/3 respectively. That is, the code starts with an if statement to check if the input format is valid; following with a for loop calculating the value of each minimum unit and summing all units up. The summed-up value is the final result to return. The MATLAB code achieving these are provided in appendix B.

Result:

By using MATLAB implementing Trapezoidal Rule and Simpson 1/3 to calculate the desired values to solve the problem, the result MATLAB return by the code provided in appendix C is shown in figure 5.

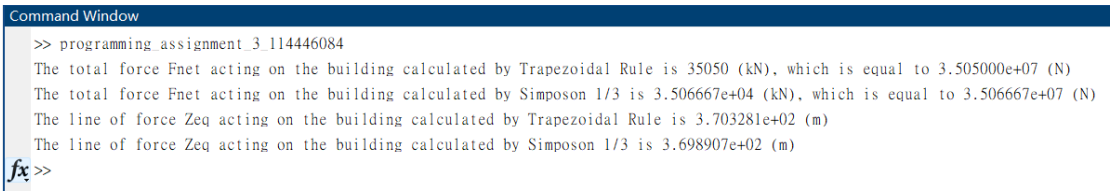


Figure 5. The result MATLAB return by the code provided in appendix C.

The total force F_{net} acting on the building calculated by Trapezoidal Rule is *35050 (kN)*, which is equal to *35050000 (N)*. The total force F_{net} acting on the building calculated by Simpson 1/3 is *35066.67 (kN)*, which is equal to *35066670 (N)*. The line of force Z_{eq} acting on the building calculated by Trapezoidal Rule is *370.3281 (m)*. The line of force Z_{eq} acting on the building calculated by Simpson 1/3 is *369.8907 (m)*. Table 4 represents these result in a friendlier format.

	Trapezoidal Rule	Simpson 1/3
F_{net} *(N)*	35050000	35066670
Z_{eq} *(m)*	370.3281	369.8907

Table 4. The final result for the force and the line of force on the building.

Discussion:

Compare the F_{net} with the curve in figure 1, the results obtained by both methods are physically possible since it is between *0* to *60000000 (N)*. Similarly, the Z_{eq} obtained from both methods are all possible. Figure 1 shows that the total height of the skyscraper is *600 (m)* and the distributed forces are generally increasing with the height; That is, the final result of Z_{eq} should range from *300* to *600 (m)*. The Z_{eq} obtained from both methods are both on this range.

The total panels for both approaches are the same; the total number of panels used for both results are 24, which makes them comparable with other. The problem states that the forces are measured during a storm. It is more likely for the force curve on the building being continuous and smooth, which means a polynomial fitting are more likely to be more accurate than a linear fit. In conclusion, the results from Simpson 1/3 are may be closer to the actual condition. From this statement, we could define the approximate error of the results from Trapezoidal Rule to be

$$\varepsilon_a = \frac{\text{result from Trapezoidal Rule} - \text{result from Simposon 1/3}}{\text{result from Simposon 1/3}} \times 100\% \qquad (5)$$

That is, the approximate error of F_{net} from Trapezoidal Rule is

$$\varepsilon_{a,F_{net}} = 0.0475\% \qquad (6)$$

and the approximate error of Z_{eq} from Trapezoidal Rule is

$$\varepsilon_{a,Z_{eq}} = 0.1183\% \qquad (7)$$

Remember the data input for both methods shown in table 3 comes from eye estimating the curve on figure. The uncertainty of the estimation is around 10%. It would be safe to conclude that for the 24 segments calculation we took, the results for both methods are good enough to represent the actual problem since equation (6) and (7) are all less than 10%.

Conclusion:

The force and the line of force on the building is calculated using Trapezoidal Rule and Simpson 1/3 using 24 panels. The total force F_{net} acting on the building calculated by Trapezoidal Rule *35050000 (N)*. and the value is *35066670 (N)* for Simpson 1/3. The line of force Z_{eq} acting on the building calculated by Trapezoidal Rule is *370.3281 (m)*, and the value is *33741670 (N)* for Simpson 1/3 *369.8907 (m)*. From the natural of the problem, results obtained from Simpson 1/3 is predicted to be more accurate. Most of the discrete data points used in the numerical integrations are from eye estimation with 10% uncertainty. The final difference between 2 results are all less then 1%, which means 24 panels are precise enough for the problem.

Reference:

Chapra, Steven C. and Canale, Raymond P. “Numerical Methods for Engineers.” Mcgraw-hill, eighth edition (2021)

Appendix:

A: MATLAB Code for Newton's Interpolating Polynomial

```
z=[0 50 100 150 225 300 375 450 600];
F=[0 30 40 40  50  50  60  80 100];
  n1=newton_interpolation(z,F);
  % fOfz : The force distribution curve function obtained by Newton's
  interpolating polynomial method over data points from table 1
  fOfz=@(x) F(1)+ ...
      (x-z(1))*n1(1)+ ...
      (x-z(1))*(x-z(2))*n1(2)+ ...
      (x-z(1))*(x-z(2))*(x-z(3))*n1(3)+ ...
      (x-z(1))*(x-z(2))*(x-z(3))*(x-z(4))*n1(4)+ ...
      (x-z(1))*(x-z(2))*(x-z(3))*(x-z(4))*(x-z(5))*n1(5)+ ...
      (x-z(1))*(x-z(2))*(x-z(3))*(x-z(4))*(x-z(5))*(x-z(6))*n1(6)+ ...
      (x-z(1))*(x-z(2))*(x-z(3))*(x-z(4))*(x-z(5))*(x-z(6))*(x-
  z(7))*n1(7)+ ...
      (x-z(1))*(x-z(2))*(x-z(3))*(x-z(4))*(x-z(5))*(x-z(6))*(x-z(7))*(x-
  z(8))*n1(8);

  %Newton's interpolating polynomial method
  function finite_devided_diff= newton_interpolation(x,y)
      if(size(x)~=size(y))
          error('Input size disagree')
      end
      f=zeros(size(x,2),size(x,2)+1); %init matrix
      f(:,1)=x';
      f(:,2)=y';
      %form the table from FIGURE 18.5 in the textbook and claculate the
  finite devided difference values
      for col=3:size(f,2)
          for row=1:size(f,1)
              if(row+col-2<=size(f,1))
                  f(row,col)=(f(row+1,col-1)-f(row,col-1))/(f(row+col-
  2,1)-f(row,1));
```

```
            end
        end
    end
    %return the desire finite devided difference values f[x1,x0]
f[x2,x1,x0]...f[xn,xn-1,...x0] only
    finite_devided_diff=f(1,3:size(f,2));
end
```

B: MATLAB Code for Trapezoidal Rule and Simpson 1/3

```
%Trapezoidal Rule method
function intg=trapezoidalRule(x,y)
    %input checking
    if(size(x)~=size(y))
        error('Input size disagree')
    end
    %calculate the value of each minimum unit and summing all units up
    sum=0;
    for ii=2:size(x,2)
        sum=sum+(y(ii)+y(ii-1))*(x(ii)-x(ii-1))/2;
    end
    %return the result
    intg=sum;
end

%Simpson 1/3 method
function intg=simpsonOneThird(x,y)
    %input checking
    if(size(x)~=size(y))
        error('Input size disagree')
    end
    if(mod(size(x,2),2)~=1 || size(x,2)<3)
        error('Unsuficient number of segments')
    end
     %calculate the value of each minimum unit and summing all units up
     sum=0;
     width=x(3)-x(1);
     for ii=3:2:size(x,2)
         if(width~=(x(ii)-x(ii-2)))
```

```
            error('unequal width')  % checking if width between all points
are equal
         end
        sum=sum+width*(y(ii-2)+4*y(ii-1)+y(ii))/6;
     end
    %return the result
    intg=sum;
end
```

C: The MATLAB Script used to solve the given problem

```
%Actual data points from eye checking the given curve used to solve the
problem (represents table 3 in the report)
z=0:25:600;
F=[0 15    30   38   40   40   40   43   47   50 50    50   50   52   57   60
  70   75 80    88   90   92   96   99   100];
M=[0 375 1500 2850  4000 5000    6000 7525 9400    11250 12500  13750
15000 16900 19950   22500    28000    31875 36000  41800    45000    48300
  52800    56925    60000];

%Calculation of each values
Fnet_trop=trapezoidalRule(z,F);
Fnet_simp=simpsonOneThird(z,F);
Mnet_trop=trapezoidalRule(z,M);
Mnet_simp=simpsonOneThird(z,M);
zeq_trop=Mnet_trop/Fnet_trop;
zeq_simp=Mnet_simp/Fnet_simp;

%Printing the result out
X1=sprintf('The total force Fnet acting on the building calculated by
Trapezoidal Rule is %d (kN), which is equal to %s
(N)',Fnet_trop,Fnet_trop*1000);
X2=sprintf('The total force Fnet acting on the building calculated by
Simposon 1/3 is %d (kN), which is equal to %s
(N)',Fnet_simp,Fnet_simp*1000);
X3=sprintf('The line of force Zeq acting on the building calculated by
Trapezoidal Rule is %d (m)',zeq_trop);
X4=sprintf('The line of force Zeq acting on the building calculated by
Simposon 1/3 is %d (m)',zeq_simp);
disp(X1);
```

```
disp(X2);
disp(X3);
disp(X4);
```