**ENED 1091: Homework #5**

**Due Week of March 7th at beginning of Recitation Session**

**INCLUDE UNITS WHEREVER APPLICABLE IN YOUR ANSWERS!**

**Problem 1:** The linear velocity of a piston connected to a crank of radius, r, with a rod of length, L, is given by:



ω is the angular velocity of the crank (radians/sec) and θ is the crank angle. Suppose r = 0.03 m, L = 0.5 m, and ω = 150 r.p.m.

1. Convert the angular velocity, ω, to radians per second. **Show Calculations.**

** =** 150 rev/min(2π rad/rev)(1 min/60 sec) = 5π rad/sec

1. Plug the values for r, L, and ω into the linear velocity equation. Using MATLAB, take the first derivative of velocity with respect to the crank angle, θ. Then find all valid angle values (CriticalPoints) which make the first derivative zero. **Give angles in units of degrees.**

**MATLAB Commands:**

LV = sym('-r\*w\*sin(th)-(r^2\*w\*sin(2\*th))/(2\*L)');

r = .03; L = .5; w = 5\*pi;

LV\_th = subs(LV);

der1LV = diff(LV\_th,'th');

crit\_pts = double(solve(der1LV == 0,'th'));

crit\_pts\_deg = crit\_pts\*180/pi;

**Critical Points (Angles):**

θ = 86.546⁰, 273.454⁰

1. Apply the 2nd derivative test to prove which crank angle results in a maximum linear velocity and which crank angle results in a minimum linear velocity. **Give angles in degrees.**

**MATLAB Commands:**

der2LV = diff(der1LV,'th');

th = crit\_pts;

minmax = double(subs(der2LV));

**Angle at which Velocity is a Maximum:**

θ = 273.454⁰

**Angle at which Velocity is a Minimum:**

θ = 86.546⁰

1. Plug the angles derived in part (c) into the velocity equation to find the maximum and minimum linear velocity of the piston.

**Maximum Velocity:**

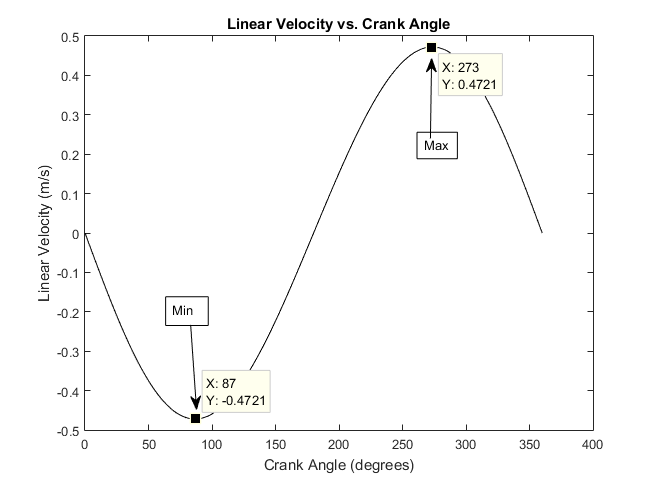
0.4721 m/s

**Minimum Velocity:**

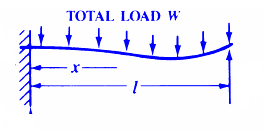
-0.4721 m/s

1. Verify your results graphically by plotting linear velocity vs. crank angle. The crank angle, θ, should go on the x-axis and vary from 0 to 360o. Mark the maximum and minimum linear velocities using the data cursor tool. Make sure your plot is titled, labeled and includes units.

**Plot of Velocity with Max and Min Points Marked:’**

****

**Problem 2:** Consider the diagram of the beam below. The beam is fixed at one end, supported at the other end, and has a uniform load of W, Newtons.



L

Modified version of the diagram from this site:

<http://www.engineersedge.com/beam_bending/beam_bending11.htm>

The stress, S (N/m2), at any point on the beam is given by:



W is the load, Z is the section modulus of the cross-section of the beam, L is the length of the beam, and x is the distance from the fixed end.

1. Suppose W = 15,000 N and Z = 0.5 m3. Find the distance, x, from the fixed end of the beam at which the stress will be minimized. Note: your answer will be a function of beam length, L.

**MATLAB Commands and/or Work:**

S = sym('(W/(2\*Z\*L))\*(L-x)\*(.25\*L-x)');

W = 15000; Z = .5;

S\_x = subs(S);

der1S = diff(S,'x');

crit\_points = solve(der1S == 0,'x');

**Distance x at which stress is minimized:**

x = 0.625\*L m

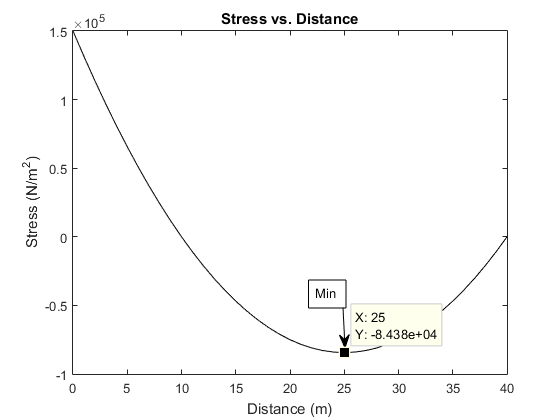
1. Assume L = 40 m. Find the distance x from the fixed end of the beam at which stress is minimized and calculate the minimum stress.

**x =** 25.0 m

**Minimum S =** -84375 N/m2

1. Verify your results graphically by plotting stress, S, on the y-axis versus distance, x. Use the data cursor to mark the point of minimum stress.

**MATLAB PLOT:**

****

**MATLAB Commands for Plot:**

dist = 0:L;

stress = W/(2\*Z\*L)\*(L-dist).\*(.25\*L-dist);

plot(dist,stress,'k-');

xlabel('Distance (m)');

ylabel('Stress (N/m^2)');

title('Stress vs. Distance');

**Problem 3:** Consider the same beam from Problem 3. The deflection, y (m), at any point on the beam is given by:



W is the load, E is the modulus of elasticity, I is the moment of inertia, L is the length of the beam, and x is the distance from the fixed end of the beam.

1. Assume W = 15,000 N, E = 5\*1010 N/m2, and I = 0.25 m4. Find all points, x, at which the deflection of the beam is either minimized or maximized. Note: x may again be a function of beam length, L.

**MATLAB commands and/or Work:**

y = sym('W/(48\*El\*In\*L)\*x^2\*(L-x)\*(3\*L-2\*x)');

W = 15000; El = 5\*10^10; In = .25;

y\_x = subs(y);

der1y = diff(y\_x,'x');

crit\_points = solve(der1y == 0,'x');

**Values of x for minimum or maximum deflection:**

x = ± (L\*(33^(1/2) + 15))/16

1. Use the 2nd derivative test to determine where minimum and maximum deflection occur.

**Value of x for minimum deflection =** (L\*(33^(1/2) + 15))/16

**Value of x for maximum deflection = -** (L\*(33^(1/2) + 15))/16

**MATLAB commands and/or Work:**

y = sym('W/(48\*El\*In\*L)\*x^2\*(L-x)\*(3\*L-2\*x)');

W = 15000; El = 5\*10^10; In = .25; %L = 40; %arbitrary L to find xmin and xmax

y\_x = subs(y);

der1y = diff(y\_x,'x');

crit\_points = solve(der1y == 0,'x');

der2y = diff(der1y,'x');

x = crit\_points;

der2y\_crit = subs(der2y);

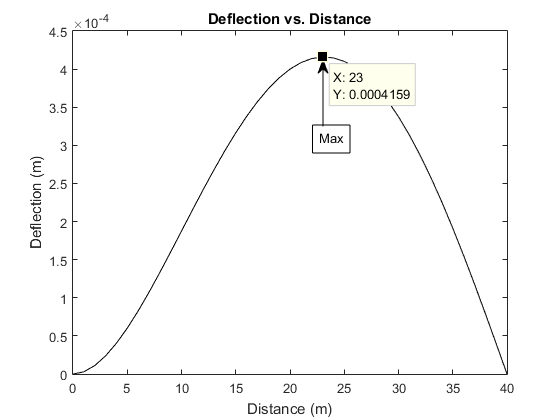
1. Assume L = 40 m. Find the point on the beam where maximum deflection occurs and calculate the maximum deflection.

**Distance x at which deflection is a maximum =** 23.1386 m

**Maximum Deflection =** 4.160e-4 m

1. Verify your results graphically by plotting deflection on the y-axis versus distance x. Use the data cursor tool to mark the point of maximum deflection.

**MATLAB PLOT:**

****

**MATLAB Commands for Plot:**

dist = 0:L;

def = W/(48\*El\*In\*L)\*dist.^2.\*(L-dist).\*(3\*L-2\*dist);

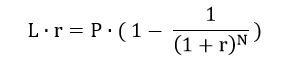
plot(dist,def,'k-');

xlabel('Distance (m)');

ylabel('Deflection (m)');

title('Deflection vs. Distance');

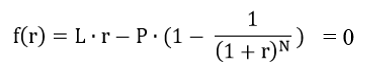
**Problem 4:** The following equation applies to a loan with monthly payments.



L is the loan amount ($), r is the monthly interest rate (in decimal, not percent), P is the monthly payment ($), and N is the number of monthly payments.

This equation cannot be solved algebraically to get a closed-form expression for r. However, the Newton-Raphson algorithm can be used to find r.

Re-write the equation as follows:



So, we are solving for the value of r that satisfies f(r) = 0.

1. Write down the Newton Raphson update equation that will be used to solve iteratively for r.

rn+1  = rn f(r)/(L – PN/(1 + r)^N+1)

1. Write a MATLAB script that will do the following:

* Prompt the user for the loan amount, L, the monthly payment, P, and the number of payments, N.
* Set the initial guess for r equal to 0.01 (i.e. a 1% monthly interest rate).
* Add a while loop to update r using the Newton Raphson algorithm as long as the absolute value of f(r) exceeds 0.0001.
* After the while loop, add fprintf statements to output the monthly interest rate and the yearly interest rate as percentages with two places behind the decimal point. ***Note: annual interest rate is simply the monthly rate times 12.***

**PASTE CODE HERE:**

L = input('Loan amount: ');

P = input('Monthly payment: ');

N = input('Number of payments: ');

r = .01; %initial guess

while abs(L\*r-P\*(1-1/(1+r)^N)) > .0001

r = r - (L\*r-P\*(1-1/(1+r)^N))/(L-P\*N/(1+r)^(N+1));

end

r = r\*100;

fprintf('Monthly interest rate: %0.2f\n',r);

fprintf('Yearly interest rate: %0.2f\n',12\*r);

1. Run your script using the values shown in the table below and fill in the table***. Note: you can easily verify whether or not your script is producing the right r-value by plugging results back into the original equation.***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Loan Amount ($)** | **Monthly Payment ($)** | **Number of Payments** | **Monthly Interest Rate (%)** | **Annual Interest Rate (%)** |
| 25000 | 500 | 60 | 0.62 | 7.42 |
| 150000 | 800 | 360 | 0.41 | 4.94 |