ME 481 - H03 Name ________________________

1) Develop a WM simulation for an in-line slider crank in a single cylinder air compressor with 2.75 inch bore, 1.97 inch stroke and connecting rod length 4.33 inch operating at constant 1000 rpm crank speed. Export the position, velocity and acceleration of the piston into a *.dta file. Be certain to start your simulation at Top Dead Center (TDC). Attach a screen shot of your WM mechanism.

R = stroke/2= 0.985 in L = 4.33 in $\left(\frac{1000\text{ rev}}{\text{min}}\right)\left(\frac{\text{min}}{60\text{ sec}}\right)\left(\frac{2\pi\text{ rad}}{\text{rev}}\right) = 104.719\text{ rad/sec}$

2) Use MS-Notepad or MS-Word to remove the header lines from the *.dta file. Write a MATLAB program to read the data from the modified *.dta file with the *load* function.

3) Program geometric exact equations in MATLAB to compute piston position, velocity and acceleration for the slider crank. Attach hard copy of your code.

4) Create three MATLAB graphs with WM and exact geometric equation results on the same graph. Use circle symbols for your WM results and a continuous line for your equation results.

a) piston position versus crank angle
b) piston velocity versus crank angle
c) piston acceleration versus crank angle

5) How many standard CFM (cubic feet per minute) of air will this compressor deliver?

CFM ___6.77 scfm___

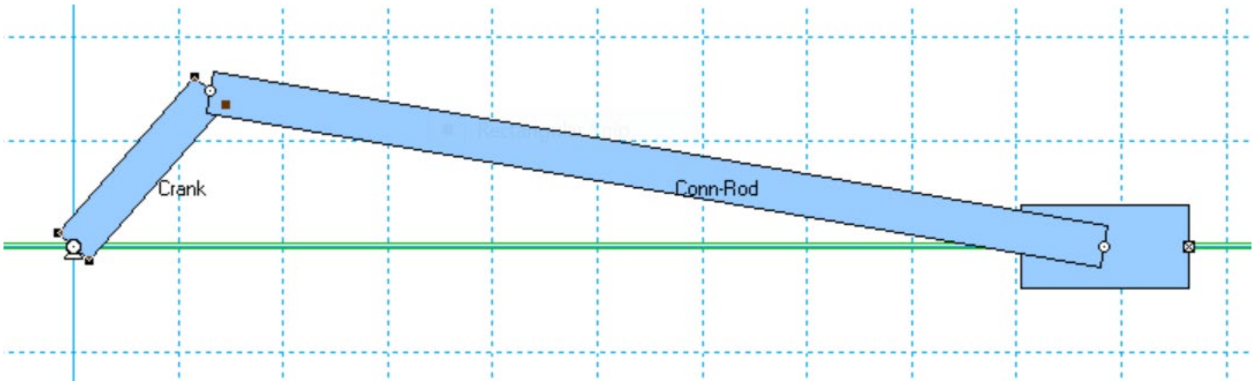
$$\text{cfm} = \frac{\pi}{4}\text{bore}^2 \times \text{stroke} \times \text{rpm} = \frac{\pi\,(2.75\text{ in})^2}{4}(1.97\text{ in})\left(\frac{1000\text{ rev}}{\text{min}}\right)\left(\frac{\text{ft}}{12\text{ in}}\right)^3$$

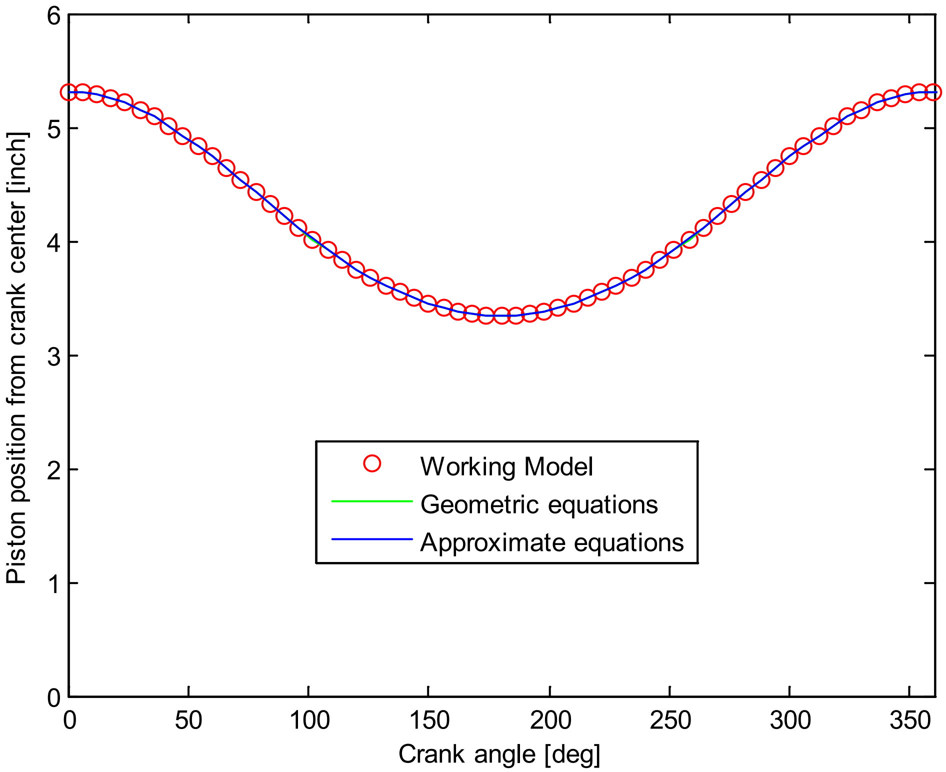
6) Save your WM file with the Measure window active and record the file size. Then open your WM file, delete the Measure window, save under a new name and record file size.

with Measure window ___471 KB___ without Measure window ___10 KB___

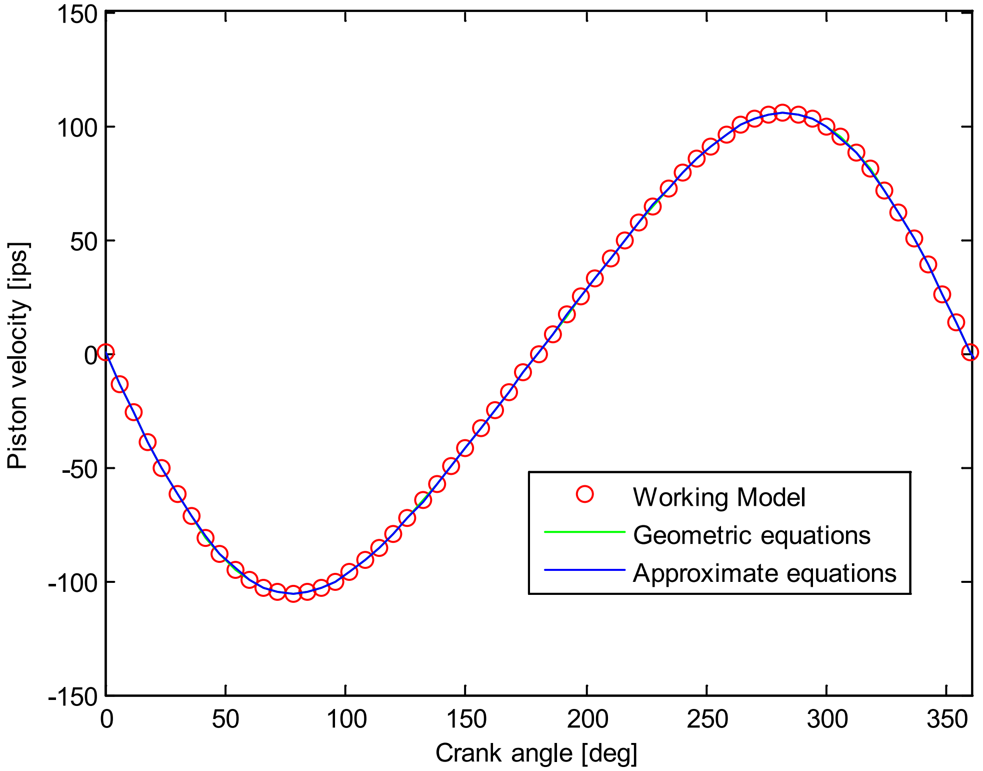
EXTRA CREDIT

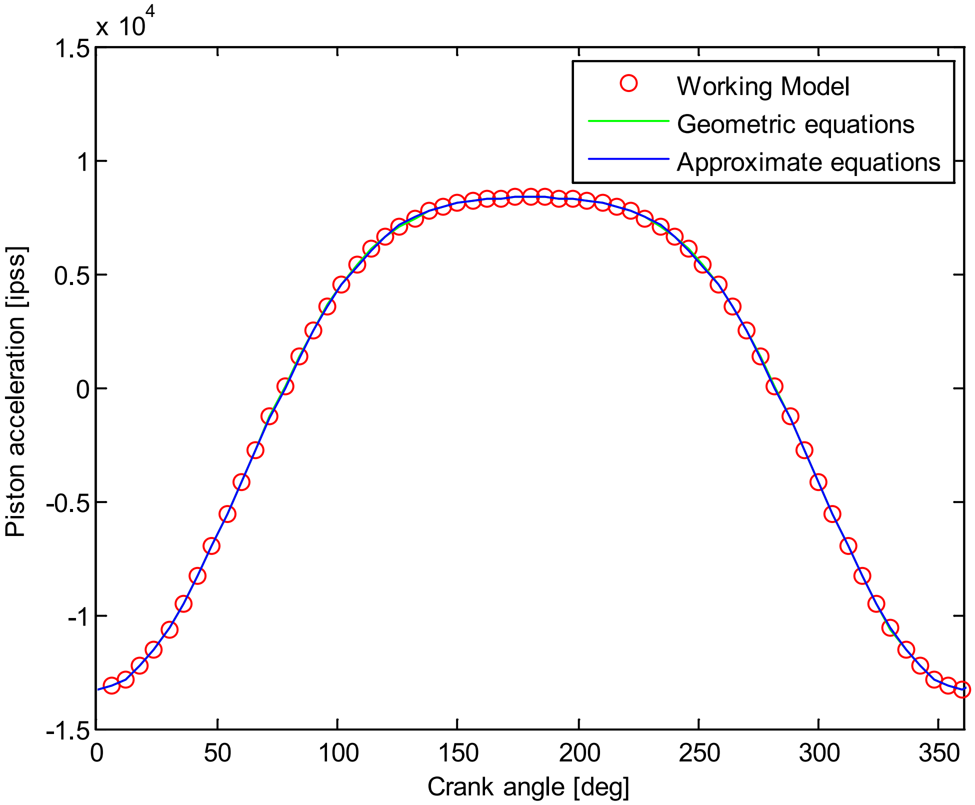
Create three additional MATLAB graphs similar to part 4) above that show the difference between exact geometric equations and binomial series approximations for this slider crank as a function of crank angle.


Crank
Conn-Rod

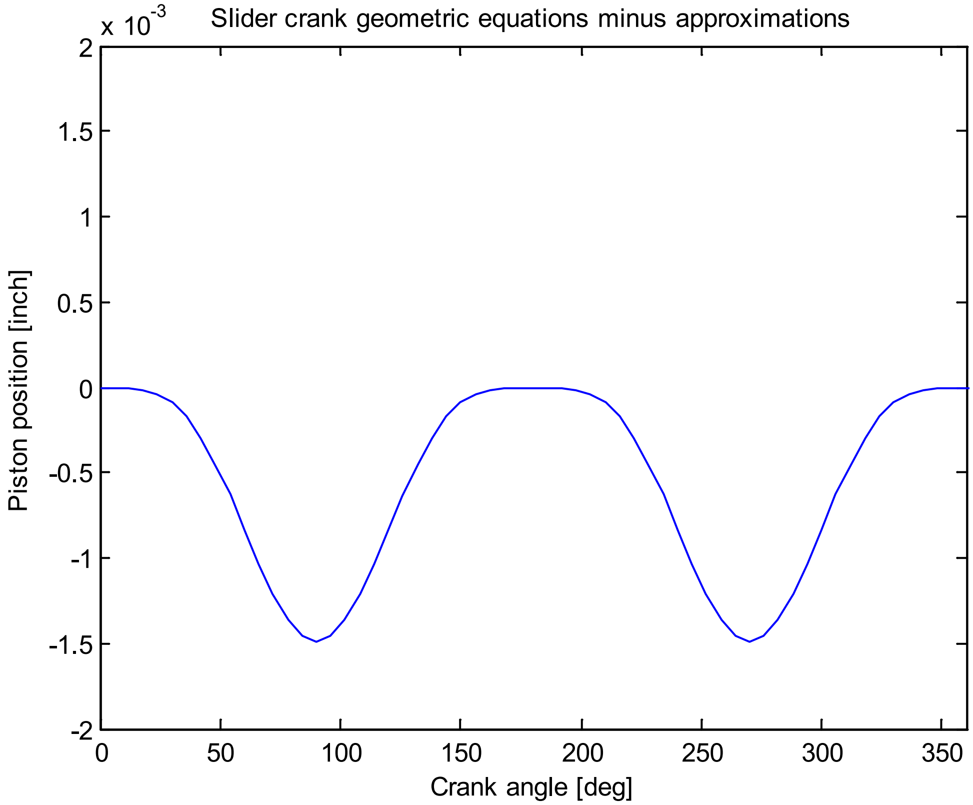

6
5
4
3
2
1
0
0
50
100
150
200
250
300
350
Piston position from crank center [inch]
Crank angle [deg]
○ Working Model
Geometric equations
Approximate equations

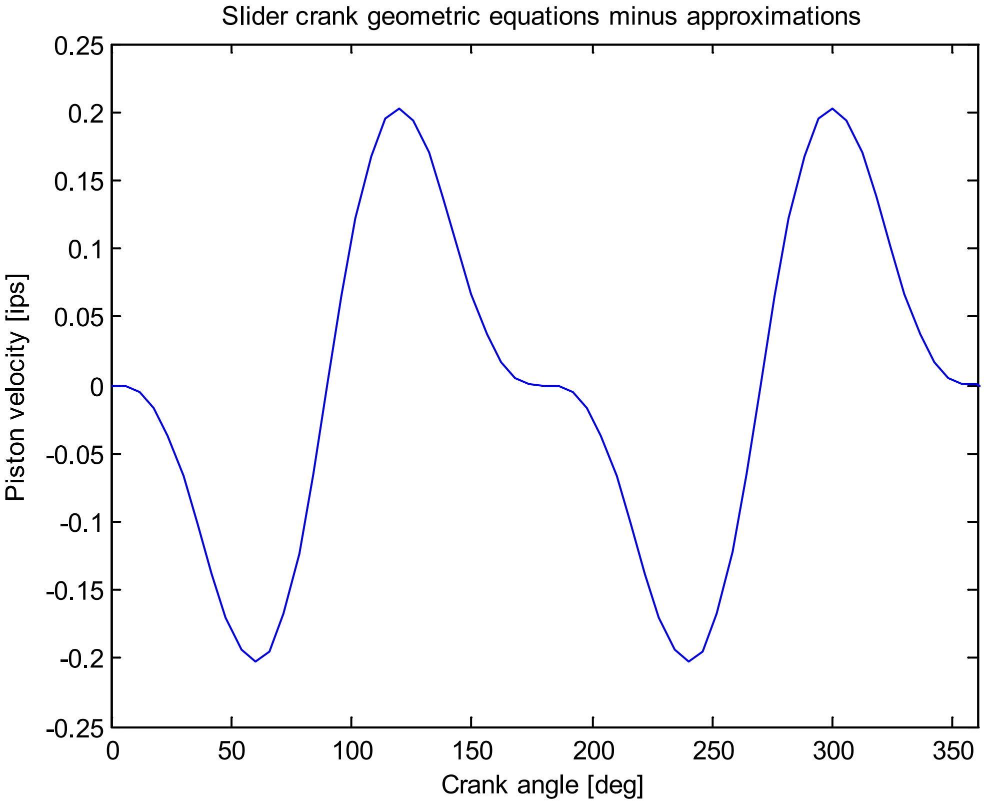
Name _______________________


150
100
50
0
-50
-100
-150
0
50
100
150
200
250
300
350
Crank angle [deg]
Piston velocity [ips]
○ Working Model
Geometric equations
Approximate equations

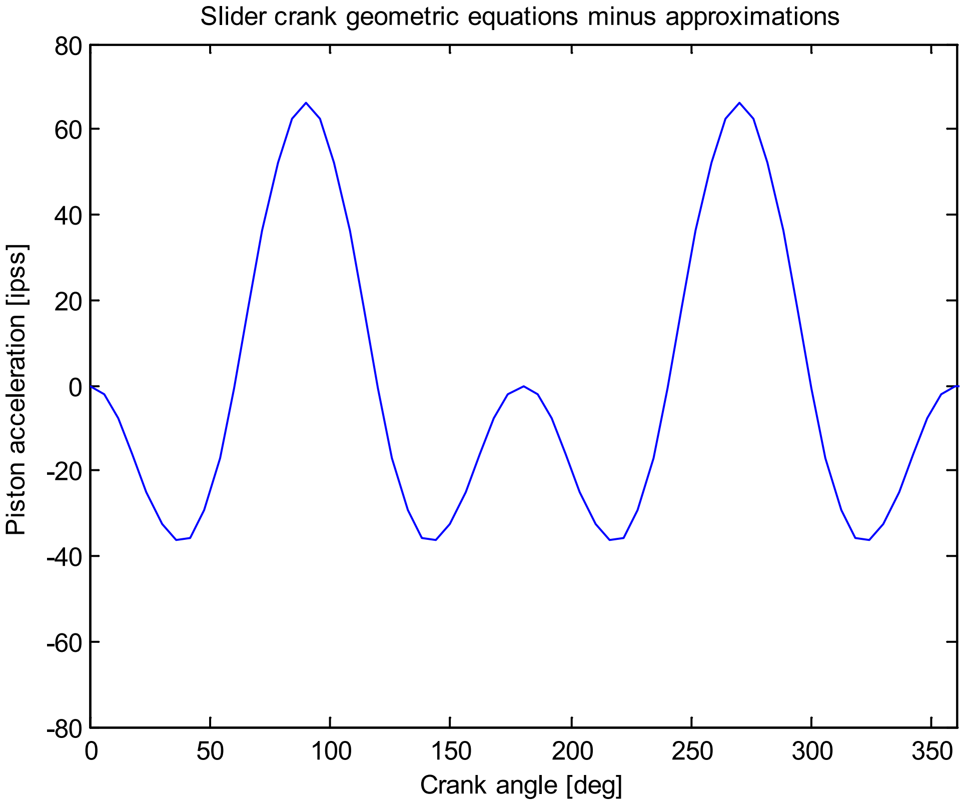

x 10^4
1.5
1
0.5
0
-0.5
-1
-1.5
0
50
100
150
200
250
300
350
Crank angle [deg]
Piston acceleration [ipss]
○ Working Model
Geometric equations
Approximate equations

Name ______________________

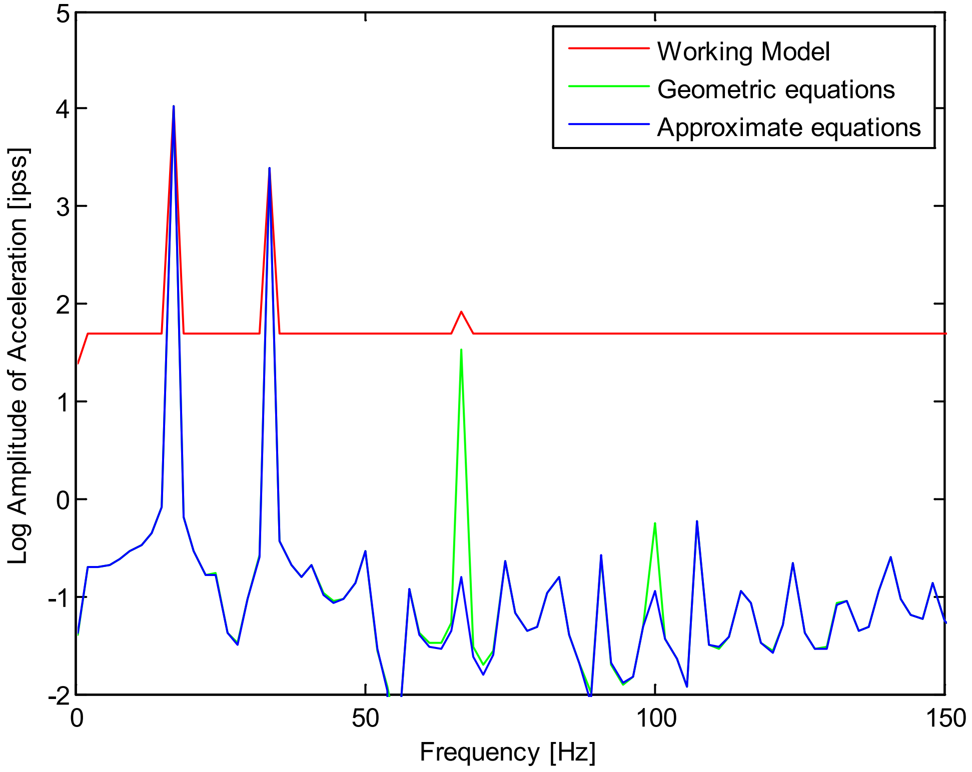

Slider crank geometric equations minus approximations
x 10^-3
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
Piston position [inch]
0
50
100
150
200
250
300
350
Crank angle [deg]


Slider crank geometric equations minus approximations
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
Piston velocity [ips]
0
50
100
150
200
250
300
350
Crank angle [deg]

ME 481 - H03 **Name ________________________**



Name ________________________


Working Model
Geometric equations
Approximate equations
Log Amplitude of Acceleration [ipss]
Frequency [Hz]
5
4
3
2
1
0
-1
-2
0
50
100
150

```
% h03.m - ME 481 H03 - read WM data for slider crank and validate
% HJSIII, 15.01.27

% general constants
d2r = pi / 180;

% mechanism constants
R = 1.97 / 2;         % crank length [inch]
L = 4.33;             % conn-rod length [inch]
w2 = 1000 *2*pi /60;  % crank speed [rad/sec]

% load WM data
load h03_cut.txt;

t = h03_cut(:,1);
th = h03_cut(:,2);
sWM = h03_cut(:,4);
vWM = h03_cut(:,5);
aWM = h03_cut(:,6);

% crank information
th_deg = th / d2r;
thdot = w2;
thddot = 0;

% geometric equations
phi = asin(  R*sin(th) / L );
sGEO = R*cos(th) + L*cos(phi);

phidot = R*thdot*cos(th) /L ./cos(phi);
vGEO = -R*thdot*sin(th) - L*phidot.*sin(phi);

phiddot = ( R*thddot*cos(th) -R*thdot*thdot*sin(th) +L.*phidot.*phidot.*sin(phi) ) ...
                 /L ./cos(phi);
aGEO = -R*thddot*sin(th) -R*thdot*thdot*cos(th) -L*phiddot.*sin(phi) ...
                -L*phidot.*phidot.*cos(phi);

% binomial series approximation equations for thddot=0
sAPP = R*cos(th) +L*( 1 -R*R/L/L/2*sin(th).*sin(th) );
vAPP = -R*thdot *( sin(th) +R/L/2*sin(2*th) );
aAPP = -R*thdot*thdot *( cos(th) +R/L*cos(2*th) );

% plot results
figure( 1 )
  plot( th_deg,sWM,'ro', th_deg,sGEO,'g', th_deg,sAPP,'b' )
  axis( [ 0 360  0 6 ] )
  xlabel( 'Crank angle [deg]' )
  ylabel( 'Piston position from crank center [inch]' )
  legend( 'Working Model', 'Geometric equations', 'Approximate equations' )

figure( 2 )
  plot( th_deg,vWM,'ro', th_deg,vGEO,'g', th_deg,vAPP,'b' )
  axis( [ 0 360  -150 150 ] )
  xlabel( 'Crank angle [deg]' )
  ylabel( 'Piston velocity [ips]' )
  legend( 'Working Model', 'Geometric equations', 'Approximate equations' )

figure( 3 )
  plot( th_deg,aWM,'ro', th_deg,aGEO,'g', th_deg,aAPP,'b' )
  axis( [ 0 360  -15000 15000 ] )
  xlabel( 'Crank angle [deg]' )
  ylabel( 'Piston acceleration [ipss]' )
  legend( 'Working Model', 'Geometric equations', 'Approximate equations' )

figure( 4 )
  plot( th_deg, sGEO-sAPP )
  axis( [ 0 360  -0.002 0.002 ] )
  xlabel( 'Crank angle [deg]' )
  ylabel( 'Piston position [inch]' )
  title( 'Slider crank geometric equations minus approximations' )
```

```
figure( 5 )
  plot( th_deg ,vGEO-vAPP )
  axis( [ 0 360  -0.25 0.25 ] )
  xlabel( 'Crank angle [deg]' )
  ylabel( 'Piston velocity [ips]' )
  title( 'Slider crank geometric equations minus approximations' )

figure( 6 )
  plot( th_deg, aGEO-aAPP )
  axis( [ 0 360  -80 80 ] )
  xlabel( 'Crank angle [deg]' )
  ylabel( 'Piston acceleration [ipss]' )
  title( 'Slider crank geometric equations minus approximations' )

% time and frequency resolution
dt = 0.001;                            % [sec]
fs = 1 / dt;                           % [Hz]
n = length( aWM );
df = fs / n;                           % [Hz]
freq = [ 0:(n-1) ]' * df;              % [Hz]

% FFT
% MATLAB FFT scaled by 2/n - DC component scaled by 1/n
fftWM = fft( aWM ) * 2 / n;     % complex number
fftWM(1) = fftWM(1) / 2;        % [ipss]
ampWM = abs( fftWM );           % [ipss]

fftGEO = fft( aGEO ) * 2 / n;   % complex number
fftGEO(1) = fftGEO(1) / 2;      % [ipss]
ampGEO = abs( fftGEO );         % [ipss]

fftAPP = fft( aAPP ) * 2 / n;   % complex number
fftAPP(1) = fftAPP(1) / 2;      % [ipss]
ampAPP = abs( fftAPP );         % [ipss]

% plot amplitude
figure( 7 )
  plot( freq,log10(ampWM),'r', freq,log10(ampGEO),'g', freq,log10(ampAPP),'b' )
  axis( [ 0  150  -2  5 ] )
  xlabel( 'Frequency [Hz]' )
  ylabel( 'Log Amplitude of Acceleration [ipss]' )
  legend( 'Working Model', 'Geometric equations', 'Approximate equations' )

% bottom of h03
```