

Excel to MATLAB data processing pipeline example:

**A mass-spring system with damping**, with position recorded via an ultrasonic sensor and a datalogger.

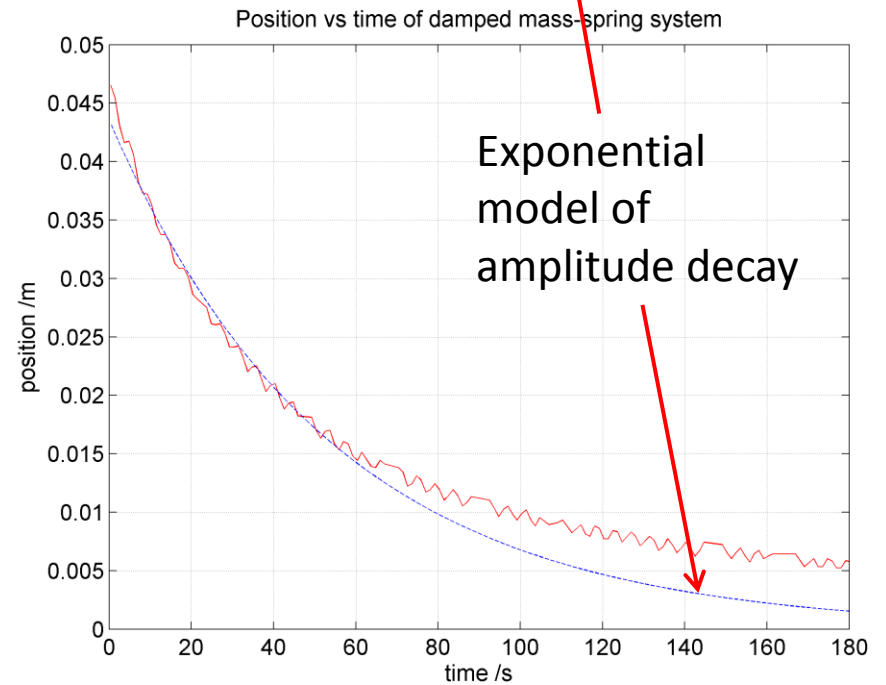
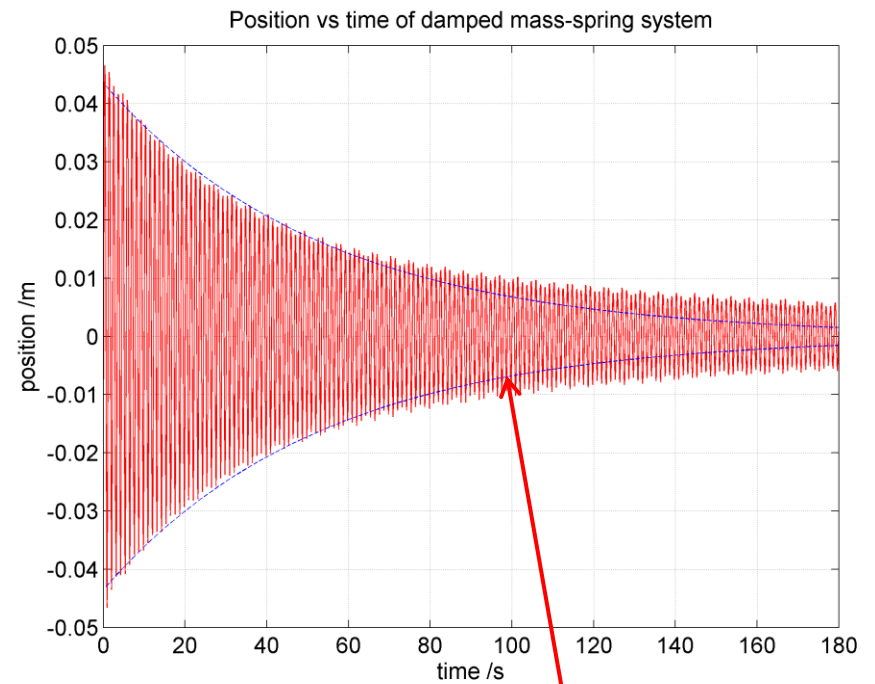
Mass-spring system  
with damping

0.200kg mass

Card damper

Ultrasonic position sensor

PASCO USB  
datalogger hub



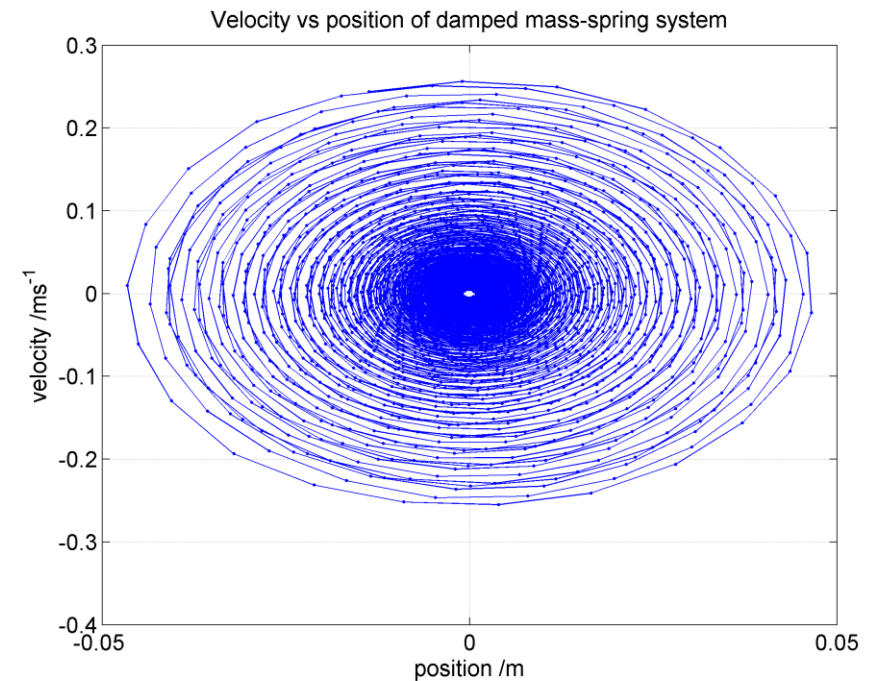
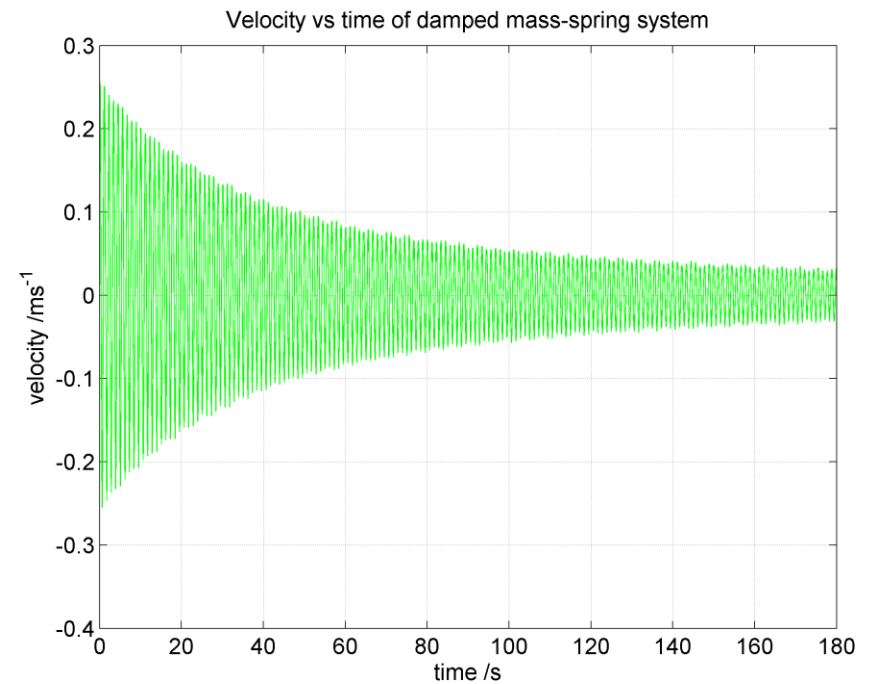
Mass-spring system  
with damping

0.200kg mass

Card damper

Ultrasonic position sensor

PASCO USB  
datalogger hub



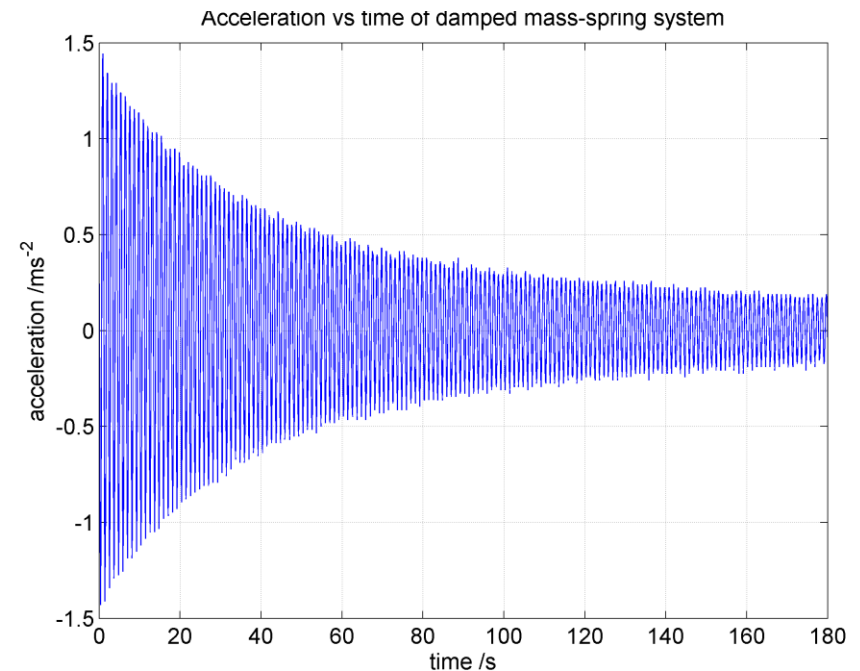
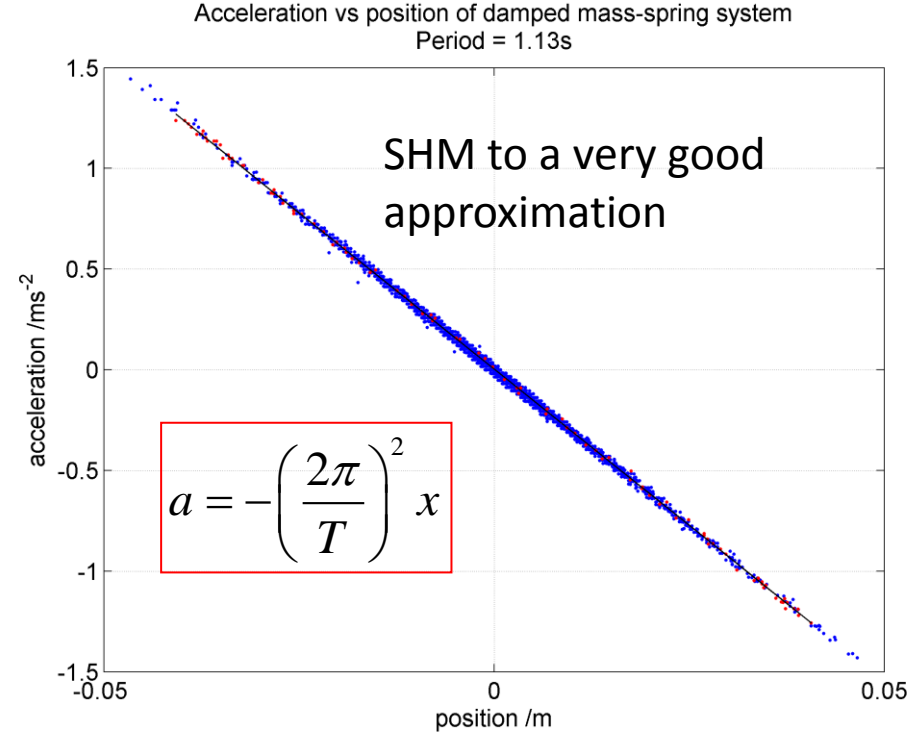
Mass-spring system  
with damping

0.200kg mass

Card damper

Ultrasonic position sensor

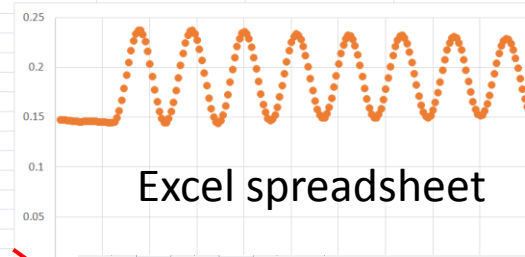
PASCO USB  
datalogger hub



# Data processing pipeline for mass-spring system

Datalogger

	A	B	C	D
1	Position (m)	Velocity (m/s)	Acceleration (m/s <sup>2</sup> )	Time (s)
2	0.1479	1.48E-05	-0.0169	0.1
3	0.1479	-0.0017	-0.0516	0.15
4	0.1478	-0.0051	-0.0519	0.2
5	0.1474	-0.0069	-0.0176	0.25
6	0.1471	-0.0069	0.0343	0.3
7	0.1467	-0.0035	0.0522	0.35
8	0.1467	-0.0017	-0.0172	0.4
9	0.1465	-0.0052	-0.0181	0.45
10	0.1462	-0.0035	0.0861	0.5
11	0.1462	0.0034	0.0695	0.55
12	0.1465	0.0035	-0.0172	0.6
13	0.1465	0.0017	-0.0173	0.65
14	0.1467	0.0017	4.41E-04	0.7
15	0.1467	0.0018	-0.0512	0.75
16	0.1469	-0.0034	-0.1038	0.8
17	0.1464	-0.0086	-0.0351	0.85
18	0.146	-0.0069	0.0515	0.9
19	0.1457	-0.0035	0.0692	0.95



```

1 % Vertically mounted three-springs-in-series are attached to a 0.2kg mass
2 % and sellotaped to a card diary to increase air resistance.
3 % Position of the mass is determined via a PASCO ultrasonic transducer,
4 % and data is logged at 20Hz using the Capstone system. Capstone
5 % automatically determines velocity and acceleration via some form of
6 % numerical differentiation.
7 function damped_mass_on_spring_analysis
8     fsize = 14; %FontSize for graphs
9     tstart = 0; tend = 60*3; %Time interval
10
11 %Load data
12 [num,tst,row] = xlsread('Damped mass on spring.xlsx');
13
14 %Ignore first few data points (oscillation hasn't started yet)
15 num = num(50:end,:);
16
17 %Determine position, velocity, acceleration, time in SI units
18 t = num(:,4); t = t - t(1);
19 x = num(:,1); v = num(:,2); a = num(:,3);
20
21 %Determine equilibrium height from the average position, and shift x
22 %such that equilibrium displacement is zero.
23 x = x - mean(x);
24
25 %Restricted data set (i.e. where effect of damping can be ignored)
26 aa = a(100:200); tt = t(100:200); xx = x(100:200);
27
28 %Determine line of best fit between a and x to determine period
29 [m,c,yfit,r] = lfit(xx,aa); T = 2*pi/sqrt(-m);
30
31 %Find peaks and plot these to determine exponential envelope
32 [tp,xp] = peakfindergeneral(t,x); N = length(xp);
33
34 %Linearize exponential envelope. Choose which set of data you want to use
35 %for this computation. NOTE THE ENVELOPE IS NOT EXPONENTIAL IN THE SAME WAY
36 %DURING THE WHOLE DECAY, SINCE THE DRAG FORCE WON'T BE LINEAR WITH SPEED.
37 i = find(xp>0); i = i(1:50);
38 Y = log(xp(i)); X = tp(i);
39
40 %Determine exponential envelope in the form x = (+/-) A*exp(-k*t)
41 [m,c,YFIT,r] = lfit(X,Y);

```

MATLAB

