Hook's law - Dynamics of a spring

Introduction

A spring is a simple system to study dynamics. A spring always opposes motion. If you stretch a spring, it will try to contract back to its original size. If you compress it, the spring will try to expand back. Thus, the force of the spring always opposes displacement. This is denoted by a negative sign. The force of the spring is also proportional to the displacement (change in the length). Thus, if you stretch the spring by 3 cm, the force will be half of what you get if you stretch the spring by 6 cm. This linear and opposing relationship between the force and the displacement is denoted by the following equation:

$$F = -k\Delta x$$

Where k is a constant for the spring. This is known as the spring constant or the stiffness constant. Higher the value of the spring constant, greater is the force required to change the length of the spring. This relationship that the spring follows is known as the Hook's law, named after Robert Hooke, the 17th century English scientist. Hook's law falls in a broader category of systems exhibiting linear response. Other examples are Ohm's law, which you will learn in PHYS 192, where electric voltage is a linear response of the electric current in conductors. Newton's second law is also an example of linear response, where the acceleration experienced by an object is a linear response to the force.

In this experiment we will verify the Hook's law by demonstrating the linear response between the force exerted and displacement of the spring. We will hang a spring vertically. We will attach a mass to the bottom of the spring. The force of gravity acting on the mass (also known as weight) will elongate the spring and this will result in the spring exerting an upward pull to the mass. These two forces, the upward pull from the spring $F_{\text{spring}} = -k\Delta x$ and the downward pull of the gravity will balance each other resulting in an equilibrium $mg - k\Delta x = 0$. Thus, we can from there find the value of the spring constant:

$$k = \frac{mg}{\Delta x}.$$

The linear relationship of the Hook's law implies that we expect to see a linear relationship in the experimental value of the elongation of the spring and the weight applied on the spring.

Materials

1. Two springs of different stiffness (you can make sure that the stiffness are different by pulling them).
2. Five to ten different masses.
3. A hanger to hang the masses that can be attached to the spring.

4. A ruler to measure the length of the spring.

Procedure:

1. First make sure you set up the system such that you can hand the spring vertically. It should be fastened at the upper end properly to make sure that it does not move when you put weights on the other end.
2. Once you have set this up, put the ruler vertically to measure the height of the lower end of the spring. Record this value.
3. Now put the hanger on the lower end. Make sure you measure the weight of the hanger first. Note down the value of the height of the lower end of the spring now.
4. Now one by one add the weights on the hanger, noting down the height of the lower end of the spring each time.
5. Once you have taken about 10 readings, calculate the extension of the spring by taking the difference between the height of the lower end of the spring for each mass and the height in absence of the mass. Let us call this Δl.
6. Graph the value of the Δl on the x-axis, and the mg in the y-axis. If this graph shows a linear trend, then Hook's law is verified.
7. Draw a linear-fit for the points, and compute the slope of that line. This will give you the spring constant.
8. Repeat this experiment for the second spring. Make sure to show the plot of the second spring in the same figure above.
9. Now join the two springs at the ends, and repeat the same experiment. Find the spring constant of this combined spring. Verify if the spring constant of this joined system k_{joined} is given by: ### $k_{\text{joined}} = \frac{k_1 k_2}{k_1 + k_2}$, where k_1 and k_2 are the stiffness constants of the individual springs.

Anaylysis

In [1]:
```
import numpy as np
import pylab as pl
import pandas as pd
```

In [2]:
```
g = 9.8
```

In [3]:
```
# White spring
L0 = 7
masses_white = np.array([150, 250, 350, 450, 550, 650, 750, 850, 950,
1050])/1000
Lengths_white = (np.array([46, 47.5, 49.5, 52, 54.5, 57, 59, 61.5, 64,
66.25]) - L0)/100
L_unstretched_white = (46.0 - L0)/100
delta_x_white = Lengths_white - L_unstretched_white

# Blue spring
masses_blue = np.array([50, 150, 250, 350, 450, 550, 650, 750, 850, 95
0])/1000
Lengths_blue = (np.array([47, 50.5, 53.5, 57, 60.5, 63.5, 66.5, 70, 73
.5, 76.5]) - L0)/100
L_unstretched_blue = (45.5 - L0)/100
delta_x_blue = Lengths_blue - L_unstretched_blue

# White and blue combined
masses_combined = np.array([50, 150, 250, 350, 450, 550, 650, 750, 850
, 950])/1000
Lengths_combined = (np.array([78.2, 82, 87.5, 93, 98.5, 104.5, 110, 11
5.5, 121, 127]) - L0)/100
L_unstretched_combined = (77 - L0)/100
delta_x_combined = Lengths_combined - L_unstretched_combined
```

In [4]:
```
# White spring
pd.DataFrame(np.vstack((masses_white, masses_white*g, delta_x_white)).
T,
             columns=["Mass (kg)", "Weight (N)", "Elongation (m)"])
```

Out[4]:

	Mass (kg)	Weight (N)	Elongation (m)
0	0.15	1.47	0.0000
1	0.25	2.45	0.0150
2	0.35	3.43	0.0350
3	0.45	4.41	0.0600
4	0.55	5.39	0.0850
5	0.65	6.37	0.1100
6	0.75	7.35	0.1300
7	0.85	8.33	0.1550
8	0.95	9.31	0.1800
9	1.05	10.29	0.2025

In [5]:
```
# Blue spring
pd.DataFrame(np.vstack((masses_blue, masses_blue*g, delta_x_blue)).T,
                 columns=["Mass (kg)", "Weight (N)", "Elongation (m)"])
```

Out[5]:

	Mass (kg)	Weight (N)	Elongation (m)
0	0.05	0.49	0.015
1	0.15	1.47	0.050
2	0.25	2.45	0.080
3	0.35	3.43	0.115
4	0.45	4.41	0.150
5	0.55	5.39	0.180
6	0.65	6.37	0.210
7	0.75	7.35	0.245
8	0.85	8.33	0.280
9	0.95	9.31	0.310

In [6]:
```
# Combined spring
pd.DataFrame(np.vstack((masses_combined, masses_combined*g, delta_x_co
mbined)).T,
                 columns=["Mass (kg)", "Weight (N)", "Elongation (m)"])
```

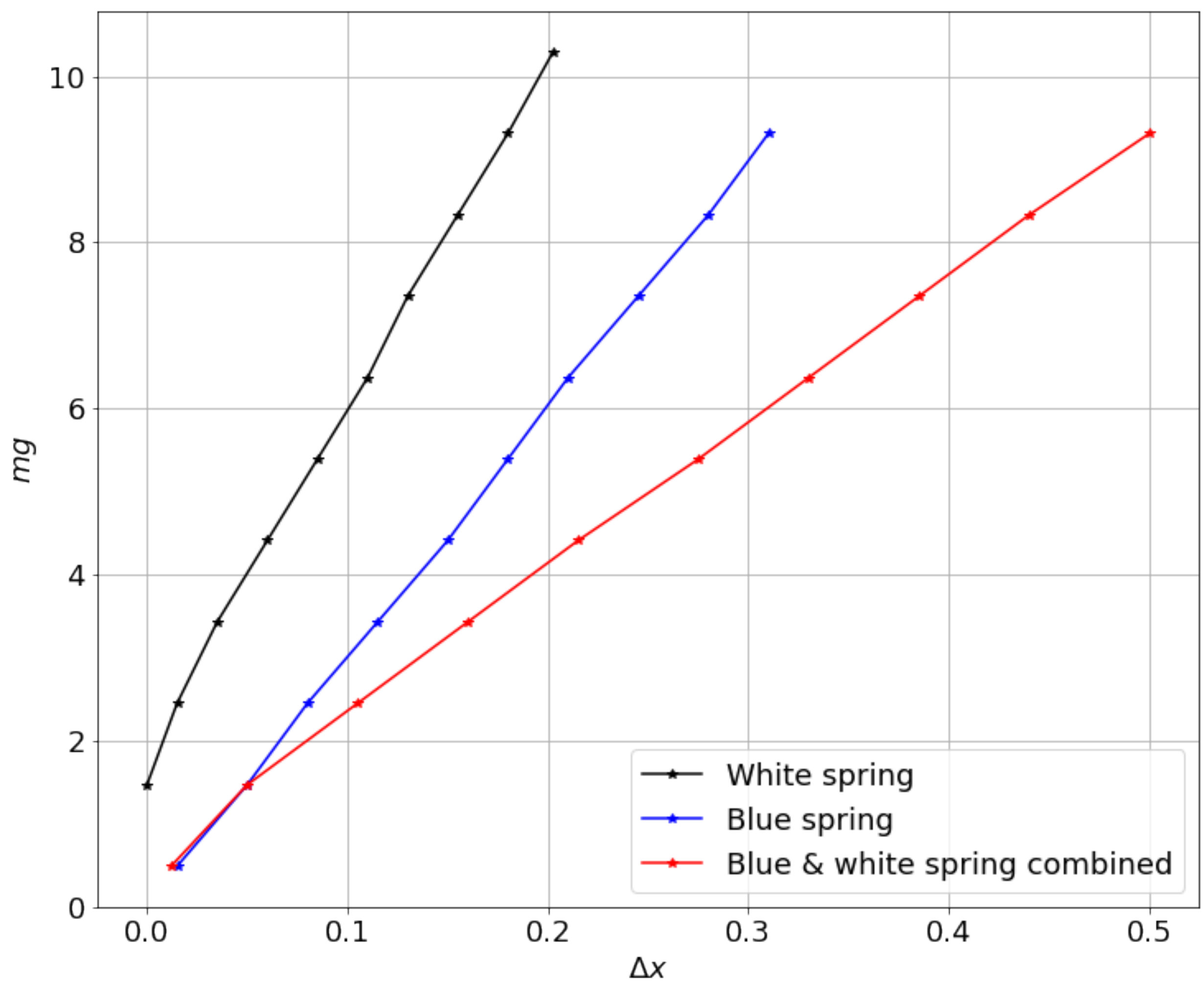
Out[6]:

	Mass (kg)	Weight (N)	Elongation (m)
0	0.05	0.49	0.012
1	0.15	1.47	0.050
2	0.25	2.45	0.105
3	0.35	3.43	0.160
4	0.45	4.41	0.215
5	0.55	5.39	0.275
6	0.65	6.37	0.330
7	0.75	7.35	0.385
8	0.85	8.33	0.440
9	0.95	9.31	0.500

In [7]:
```
pl.rcParams.update({'font.size': 18})
pl.figure(figsize=(12,10))

pl.plot(delta_x_white, masses_white*g, 'k*-', label="White spring")
pl.plot(delta_x_blue, masses_blue*g, 'b*-', label="Blue spring")
pl.plot(delta_x_combined, masses_combined*g, 'r*-', label="Blue & whit
e spring combined")

pl.legend()
pl.xlabel("$\\Delta x$")
pl.ylabel("$mg$")
pl.grid(1)
```



Linear-fit to verify the Hook's law and find the spring constants

This is only useful for you if you are doing this analysis using Python. If you are using something like Excel, then use whatever technology existss there to find the linear fit to the data.

In [8]:
```
from scipy.optimize import curve_fit
```

In [9]:
```
def fitting_func(x, a, b):
    '''
    This function allows me to fit data linearly.
    The equation below is the equation of a straight line.
    '''
    return a*x + b
```

In [10]:
```
popt_white, pcov_white = curve_fit(fitting_func, delta_x_white, masses
_white*g)
popt_blue, pcov_blue = curve_fit(fitting_func, delta_x_blue, masses_bl
ue*g)
popt_combined, pcov_combined = curve_fit(fitting_func, delta_x_combine
d, masses_combined*g)

# popt_white is just a tupple whose first element
# is the slope of the fit and the second element is the intercept See
below:
print("Slope = {}".format(popt_white[0]))
print("Intercept = {}".format(popt_white[1]))
# pcov are the covariance matrices, where are way beyond the scope of
this course. So, ignore that!
```

```
Slope = 42.33171407933516
Intercept = 1.7632408061304092
```

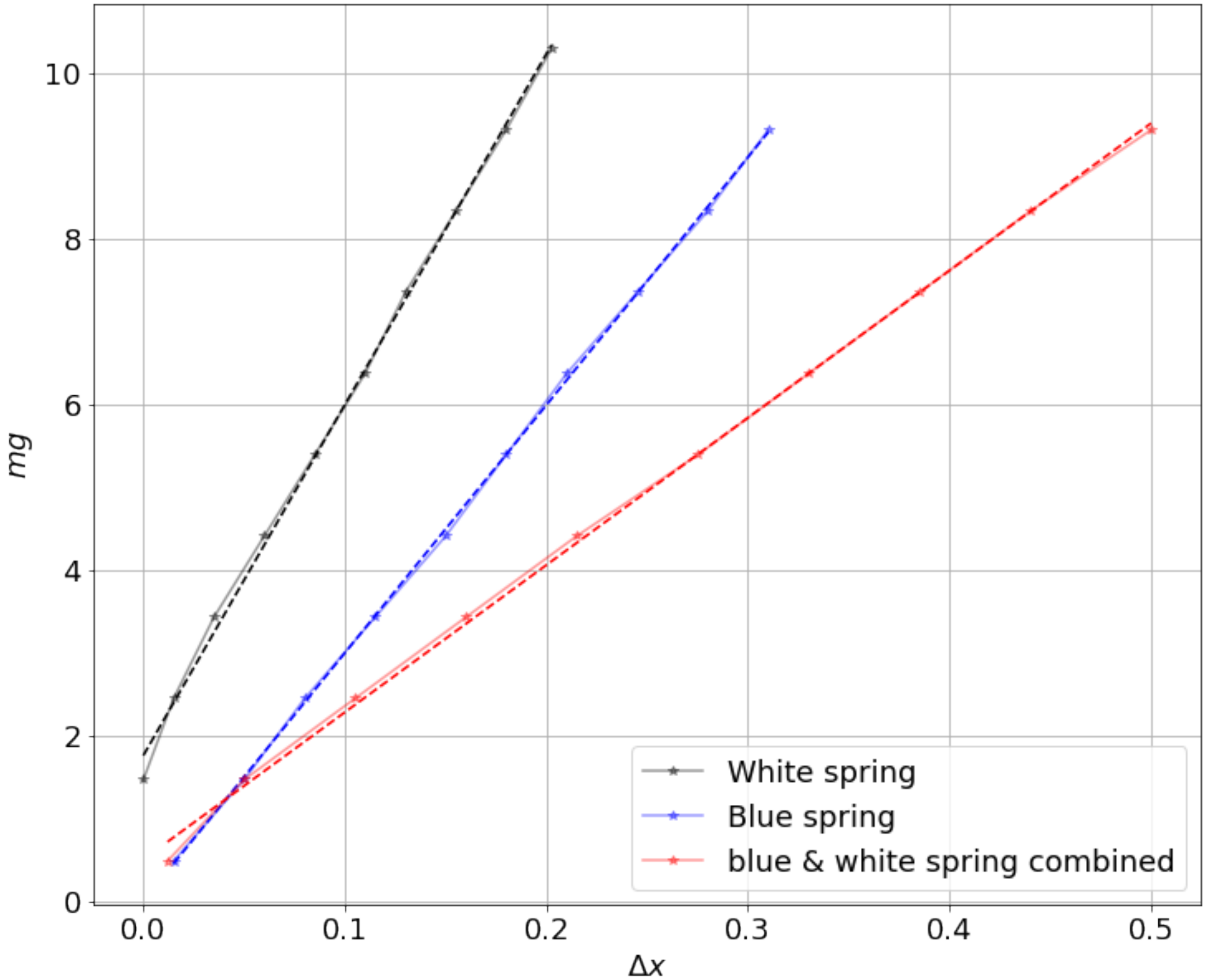
In [11]:
```
pl.rcParams.update({'font.size': 18})
pl.figure(figsize=(12,10))

pl.plot(delta_x_white, masses_white*g, 'k*-', label="White spring", al
pha=0.4)
pl.plot(delta_x_white, fitting_func(delta_x_white, *popt_white), 'k--'
)

pl.plot(delta_x_blue, masses_blue*g, 'b*-', label="Blue spring", alpha
=0.4)
pl.plot(delta_x_blue, fitting_func(delta_x_blue, *popt_blue), 'b--')

pl.plot(delta_x_combined, masses_combined*g, 'r*-', label="blue & whit
e spring combined", alpha=0.4)
pl.plot(delta_x_combined, fitting_func(delta_x_combined, *popt_combine
d), 'r--')


pl.legend()
pl.xlabel("$\\Delta x$")
pl.ylabel("$mg$")
pl.grid(1)
```



10
8
6
4
2
0
mg
0.0
0.1
0.2
0.3
0.4
0.5
Δx
White spring
Blue spring
blue & white spring combined

In [12]:
```
# Value of spring constants
#of course we already know the value of the k from the popt value abov
e. We are verifying them here.

x1 = delta_x_white[0]
x2 = delta_x_white[-1]
y1 = fitting_func(x1, *popt_white)
y2 = fitting_func(x2, *popt_white)
k_white = (y2 - y1)/(x2 - x1)
print("Value of k for the white spring = {}".format(k_white))

x1 = delta_x_blue[0]
x2 = delta_x_blue[-1]
y1 = fitting_func(x1, *popt_blue)
y2 = fitting_func(x2, *popt_blue)
k_blue = (y2 - y1)/(x2 - x1)
print("Value of k for the blue spring = {}".format(k_blue))

x1 = delta_x_combined[0]
x2 = delta_x_combined[-1]
y1 = fitting_func(x1, *popt_combined)
y2 = fitting_func(x2, *popt_combined)
k_combined = (y2 - y1)/(x2 - x1)
print("Value of k for the combined spring = {}".format(k_combined))
```

```
Value of k for the white spring = 42.33171407933516
Value of k for the blue spring = 29.908246669004686
Value of k for the combined spring = 17.765913655659343
```

In [13]:
```
k_combined_expected = k_white*k_blue/(k_blue + k_white)
```

In [14]:
```
# Deviation from expected value:
error = (k_combined_expected - k_combined)*100/k_combined_expected

print("Deviation from expected value = {}%".format(round(error, 2)))
```

```
Deviation from expected value = -1.37%
```

Extra-credit:

1. Calculate the uncertainty in the experiment using the calculus-based error analysis, like you did for the projectile motion. (5 points)
2. Make the shaded-region plots for the uncertainty for all the mg vs Δx graphs (5 points).
3. Derive in your report how $k_{\text{joined}} = \frac{k_1 k_2}{K_1 + k_2}$ (5 points)

Dicussion and Conclusion

Discussion and conclusion must be in line with what I have asked for earlier. I will not give you points in this section for simply restating what you did in the experiment.

In [ ]:
```

```