EPJ manuscript No.
(will be inserted by the editor)

The Coors Field Effect

Physics of the baseball at a higher elevation

Thomas Farley[a]

Department of Physics, University of Dallas, Irving, TX USA

November 24, 2020

Abstract. Coors Field is a paradise on earth for heavy hitters. Since the ballpark is at a higher elevation, the ball travels further due to lower air density. Fly balls are turned into home runs and bloop singles into doubles. Not only the hitting is affected, but the pitching is too. Pitchers notice that breaking balls and fast balls do not have as much vertical or horizontal break, making it a pain to locate a pitch in the desired spot. This paper will not only analyze the data of pitch breaks and minimal exit velocity a hitter needs to hit a home run at Coors, but will provide python code that simulates its effects. By understanding the effect of Coors Field in theory and numerically, we can better understand the physics of baseball, a landmark pass time in the history of the United States.

1 Introduction

The Colorado Rockies' home stadium in Denver has puzzled many over the years. Opened in 1995, the field has been a haven for power hitters and a nightmare for pitchers. Hitters have cashed in on the opportunity to play at Coors, because the ball travels further at Coors than at any other ballpark. Physicists know this attribute has a lower air density. Coors field is exactly a mile above sea level, sitting at 5280 feet of elevation. At 5000 feet of elevation, the average air density is 0.7364 kg/m^3 (?, ?). But this value for air density was calculated with a very low temperature. Temperature can affect the trajectory of an object, especially a baseball, due to its effects on air pressure and air density.

N.J. Giordano et al. presents an equation for computing air densities at higher altitudes along with different temperatures. (?, ?)

$$\rho = \rho_0 (1 - \frac{ay}{T_0})^{\alpha}, \tag{1}$$

where a is equal to 6.5×10^{-3} K/m, y is the altitude, T_0 is temperature at sea level, and α is roughly equal to 2.7. Using this equation, we can find the air density at a given altitude. The average temperature during games at Coors Field is 73.6 F (?, ?), 296.26 K. Coors Field is 5200 ft above sea level, 1584.96 m. According to Meyer and Bohn, ρ_0, taking average relative humidity into account, is roughly equal to 1.225 kg/m^3 at sea level (?, ?). This is only an estimate of the air density at different levels of altitude, for it does not take into account relative humidity as a variable. Baseball fans know that Great American ballpark (home of the Cincinnati Reds) also sees a lot of long balls. The city of Cincinnati has very humid weather often; Thogersen et al show that air density lowers in value the more humidity there is (?, ?).

Not to mention the drag force on the ball would be heresy. Giordano et al. once more come in handy, for they estimated that in relation to the air density at a given altitude, the drag force at that altitude is given by the following equation: (?, ?)

$$F_{drag}* = \frac{\rho}{\rho_0} F_{drag}, \tag{2}$$

F_{drag} is the drag force at sea level, and ρ_0 is the air density at sea level as noted previously. The drag coefficient of a baseball changes at every level of velocity. Giordano et al. fit a curve to data and found that the drag coefficient is defined by the equation,

$$C_d = 0.0039 + \frac{0.0058}{1 + e^{\frac{v - v_d}{\Delta}}}, \tag{3}$$

[a] tfarley@udallas.edu

where v_d is estimated to be roughly 35.00 m/s and Δ is estimated to be roughly 5.00 m/s.

The Magnus effect on the baseball must also be considered. The ball experiences a force of lift, which, for some pitchers, makes the ball rise due to spin. Just like the coefficient for drag, there is a coefficient for lift. The coefficient of lift, Nathan found, is nearly equal to and certainly proportional to the spin factor of the baseball (S), which is a function of the Reynolds number (?, ?). For the sake of keeping things simple, we will assume that $C_L = S$, which Nathan quotes as being roughly consistent with the data. He then defines the magnus force and the coefficient of lift with the following two equations respectively:

$$F_M = \frac{1}{2} C_L \rho A v^2 \tag{4}$$

$$C_L = \frac{R\omega}{v} \tag{5}$$

In these equations, A is the cross-sectional area of the ball, R is the radius of a baseball, and ω is the angular velocity of the ball. In addition to the drag force, we also find that the Magnus force is a function of air density. With a lower value of air density, we will expect both the force of drag and the Magnus force to be smaller. Here, the Magnus force and the air density are proportional, just as density is with the drag force. Similar to Equation (2), we can infer that $F_{lift}* = \frac{\rho}{\rho_0} F_{lift}$.

With these equations, we can begin to derive our motion equations.

$$\vec{a} = -k_D v \vec{v} + k_C \vec{\omega} \times \vec{v} - \vec{g}, \tag{6}$$

where $-k_D = \frac{C_D \rho A}{2m}$ and $-k_L = \frac{C_L \rho A}{2m}$. With this equation, we can use the Euler method to solve for the components of velocity and distance. The components of distance traveled for our baseball will be the following:

$$\frac{dv_x}{dt} = -k_D v v_x + k_L \omega (v_z sin(\phi) - v_y cos(\phi)) \tag{7}$$

$$\frac{dv_y}{dt} = -k_D v v_y + k_L \omega v_x cos(\phi) \tag{8}$$

$$\frac{dv_z}{dt} = -k_D v v_z - k_L \omega v_x sin(\phi) - g \tag{9}$$

Given these equations, we can use the Euler method to solve for the trajectory for both a home run and a pitch type. I will model the home run after the hitting style of current Rockies third baseman Nolan Arenado and model the pitches and pitch types after Kyle Freeland, Colorado's hotshot pitcher.

2 Procedures

In our python code, we define our equation for air density at a given altitude (1) and state our initial values. At an elevation of 5200 ft with a 50 F temperature, we find that the estimated air density of Coors Field on an average day is 0.976 kg/m^3. The altitude in terms of meters is 1584.96 m. The python code allows one to input the game time temperature in order to get the hypothetical air density. With this data, we can plot a simulation of the effects of air density on pitches and home runs. But before that, let us look at the data of the effects of Coors field on the pitchers.

2.1 Pitching Effects

Kyle Freeland has a career ERA of 4.17, very respectable for a pitcher who has spent his whole career thus far at Coors (?, ?). His home ERA is 4.35 and his away ERA is 3.98. Hitters score more runs at home off Freeland than they do on the road.

2.1.1 Pitcher Data Analysis

We collect data from Brooks Baseball's PITCHf/x data-site. The data of average horizontal and vertical break per game for his fastball are used. The horizontal break is the horizontal distance (m) from the release point and the vertical distance is the vertical distance (m). We separate the data into his average breaks at home and his average breaks on the road. Figure 1 shows his average breaks for the two cases:

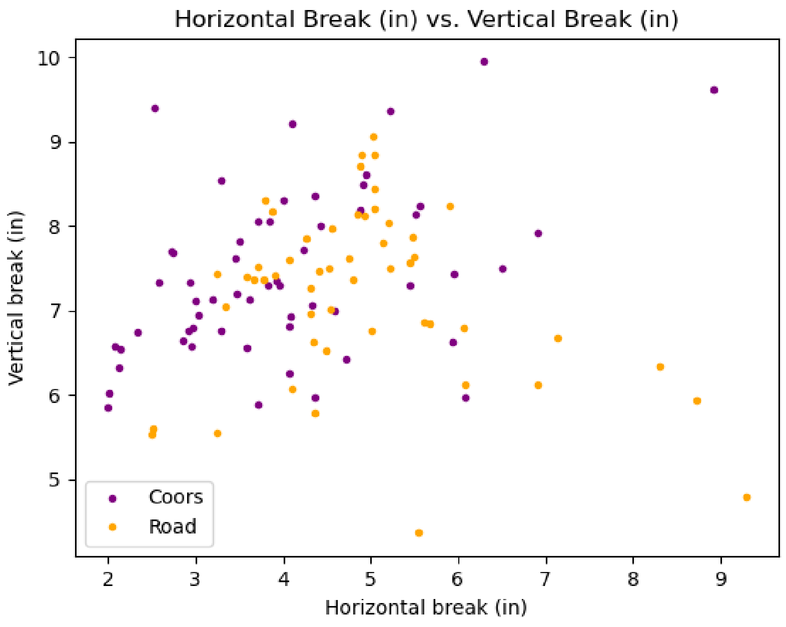

Fig. 1. Graph plots the breaks in units of inches

	Velocity (m/s)	Spin Rate (rpm)	Axis angle $(°)$
Fastball	41.13	2451	315
Curveball	35.76	2444	135
Slider	38.45	2473	180
Changeup	38.45	1412	0

Table 1. Values for velocity, spin rate, and axis angle for each type of pitch.

2.1.2 Pitching Simulation

A pitching simulation was also constructed using python in attempts to illustrate the difference between types of pitches. The simulation is modeled after Freeland's pitching style; he is a left handed pitcher with a fastball release angle at 315° (see Fig 1). Freeland has six types of pitches in his repertoire, but we only included the data from four: 4-seam fastball, curveball, slider, and changeup. The table below 2.1.2 gives the different average velocities, spin rates, and angle of axis for each type of pitch. (?, ?) The functions for k_D and k_L, as shown in Equation 6, are defined as functions of velocity (v), and velocity and angular velocity (ω) respectively. For the initial values for the function of k_L, I put $A = 0.00442$ (the cross sectional area in meters of a major league baseball, $m = 0.145$ (mass of the baseball in kg, and $R = 0.0375$ (the radius of a baseball). Following that, we define two different functions, one for the trajectory of the ball at Coors, and the other for the trajectory of the ball under conditions at sea level. We use the Euler method to solve for the trajectory traveled by the baseball using equations 7, 8, and 9. For the initial values, we input the starting position at point (0,0,1.8), where 1.8 is the release point height of Freeland in meters. $g = 9.81m/s^2$ as it should. There are separate values for each type of pitch, so as to factor in the different angular velocities, velocities, and spin rates. Graphs the trajectories of the same type of pitch for Coors and a ballpark at sea level are constructed.

For each figure, Coors is plotted in the color purple, and the average sea level ballpark is plotted in orange. All pitches except one have a release angle of 0° from the horizontal (θ). The curveball breaks downward too much, so I input an angle of 1° for its release.

2.2 Hitting Effects

Nolan Arenado may be the best Rockies player since their addition as an expansion team 1993. The California native has racked up 235 home runs and a .541 career slugging percentage in 8 seasons with the Rockies. Arenado's vast sample size of home runs will assist in illustrating the distance traveled by the long ball at Coors.

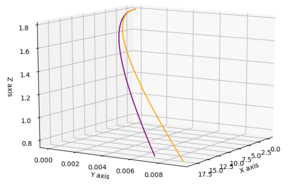

Fig. 2. This is the plot of the fastball.

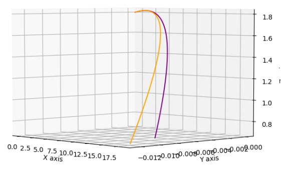

Fig. 3. This is a simulation of a curveball.

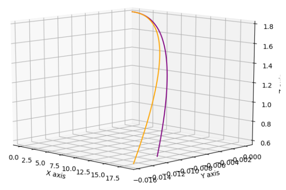

Fig. 4. Here is the trajectory of a slider.

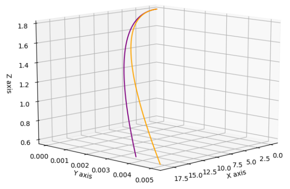

Fig. 5. This is the trajectory of a changeup.

2.2.1 Hitter Data Analysis

In the code, we load data using numpy (?, ?). Two sets are loaded, one for home stats and one for road stats. The data set has values of exit velocity (mph), home run distance (ft), and launch angle ($^\circ$) for each home run Arenado has sent into orbit. On Figure 6, exit velocity is on the x-axis and home run distance is on the y-axis for both sets of data; Coors Field home runs were plotted in a different color than away home runs. Afterwards, we import stats from scipy in attempts to plot a line of best fit for home and away stats each. The results were surprising. See Fig. 6.

2.2.2 Hitter Simulation

With the data imported from Baseball Savant, we calculate the average launch angle for both away and Coors home runs, as well as the average exit velocity for both. We create two nearly identical functions to the functions used to calculate the trajectory of Freeland's pitches and to calculate the trajectory of Arenado's home runs. The only difference is that the loop in which the Euler method was conducted has a requirement of $z > 0$ and the starting vertical point 0.6 m (where a strike over the plate could be hit for a home run). The average launch angle for both home and away came out to be 29° and the average exit velocity for both home and away surprisingly were the same as well, coming out to be 102 mph. These values are the initial conditions for the angle from the horizontal (θ) and velocity (v). The exit velocity in m/s is 45.66. According to Baseball Savant, Arenado's spin rate (angular velocity (ω)) is 2337 rpm. The trajectories were both plotted. See Fig. 7.

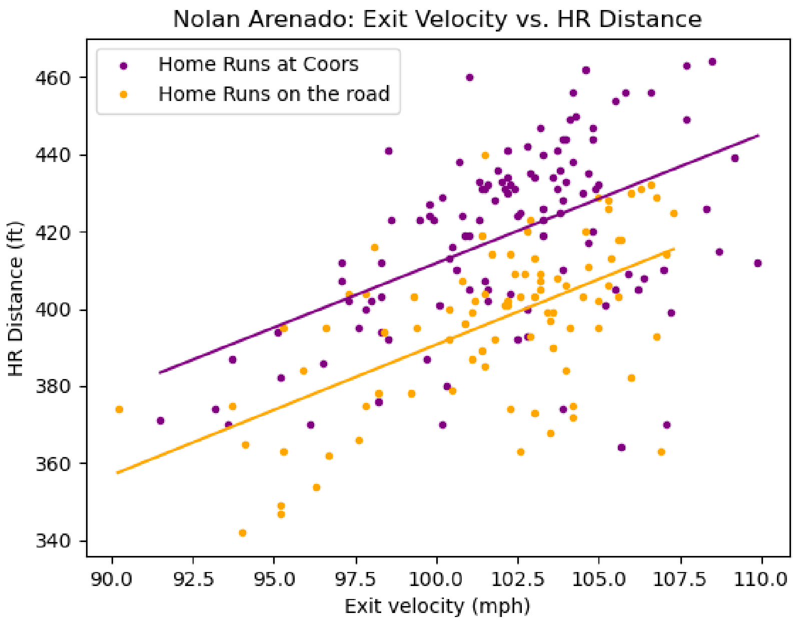

Fig. 6. Linear regression lines of Coors Field and away fields data for Arenado's home runs.

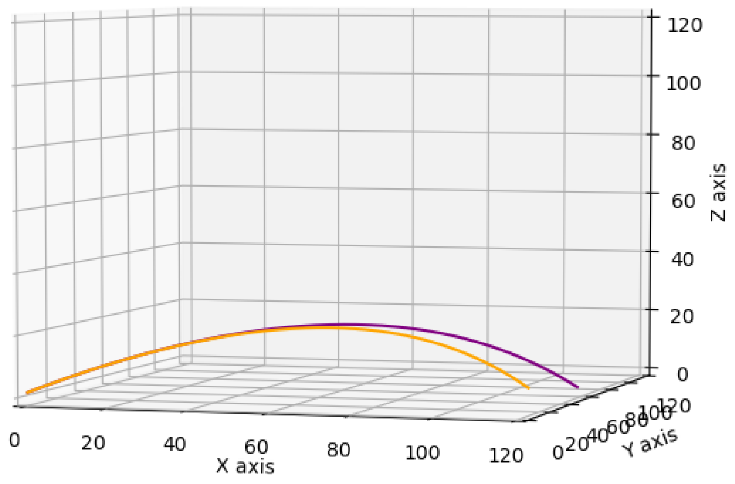

Fig. 7. Coors is plotted in purple, away is plotted in orange.

3 Results

3.1 Pitching Results

Oddly enough, the data shows that Freeland's fastball has virtually no difference in vertical break. There was a 1 inch difference in horizontal break, however. See Table 2. With the python code, the difference in the final vertical values for both trajectories as well as the final horizontal values was calculated. See Table 3.

	Vertical Break (inches)	Horizontal Break (inches)
Coors	7.2	3.9
Away	7.2	4.9
Difference	0	1

Table 2. Freeland's average breaks for vertical and horizontal at Coors and away parks.

	Difference in Vertical Break (inches)	Difference in Horizontal Break (inches)
Fastball	-0.5	0.1
Curveball	-1.2	-0.1
Slider	-0.9	-0.1
Changeup	-0.9	0.0

Table 3. Values calculated using the simulation for each pitch. Calculated as $AwayBreak - HomeBreak$.

3.2 Hitting Results

The linear regressions for Fig. 6 resulted in the following two equations:

$$y_{home} = 3.33x + 78.27 + \epsilon \tag{10}$$

$$y_{road} = 3.39x + 52.16 + \epsilon \tag{11}$$

The difference of the averages of home run distance came out to be $22.25ft$ for Arenado's home runs.

We calculate the home run distance of the simulations by computing the magnitude of the sum of the squares of the last two values for depth and length. The average HR distance at Coors according to the simulation came out to be 437.5 ft. The average HR distance for a HR hit on the road is 397.9 ft. Again, these values were calculated using the same initial conditions of hypothetical average Nolan Arenado at bat. The added distance at Coors field came out to be 39.63 ft, according to the simulation.

4 Discussion

It is interesting to note the difference in horizontal break among Freeland's fastball at home, away, and lack thereof of his vertical break. The reason for this is likely because Freeland pitches at a three-quarter slot, or nearly a side-arm angle. This would change the axis of angular velocity for each of his pitches. The values for angular velocity as shown in Table 1 were hypothetical for your average southpaw, but not catered to Freeland's exact pitching style.

The difference in horizontal break for home and away according to the simulation is nearly non existent for each pitch. The vertical break, however, depicts a bit more variance. Each pitch has more vertical break at Coors than at any other location. This is likely due to the magnus effect being at a lower level at Coors than at other ballparks. This makes sense according to the concept behind equation 2. This might be another reason why we see many home runs fly out of Coors; if pitchers use the deceptive characteristics of their pitches to their advantage and those techniques are muzzled by the higher elevation, then hitters tend to see the ball a lot better. When the hitters have better visibility, there will be more balls put in play and thus a higher chance of there being a home run added on to the effect that balls already fly much further.

The added distance for Nolan Arenado's home runs at Coors is 22.25 ft, which is about half of the added distance that our simulation states is added. The actual is less than the hypothetical due to a couple of reasons. The Rockies put their game balls in a humidor in order to make them slightly heavier and larger (?, ?). Since they are larger and heavier, this would mean they create more drag according to equation 6. Since there is more drag, the magnus effect is increasingly being counteracted. Therefore, the net force in the vertical direction is mostly due to the drag force and gravity. Also, the spin axis of the angular velocity of the home run likely varies by a tiny margin compared to what we input as an initial condition.

There is much room for error in the simulation. Again, it only accounts for changes in air density, but not humidity and air viscosity. The simulation, as already stated, does not take into account the effect of the humidor nor the exact pitching and hitting style of Freeland and Arenado respectively.

5 Conclusion

The effect of Coors Field is blatantly obvious in actuality and in theory, yet it might not be that obvious to how all variables factor together to make the ball fly further and break more vertically. Further studies will be needed to determine all the physics of Coors. Relative humidity and air viscosity are both topics that need to be analyzed and included into the study of Coors. Additionally, the baseball is not all in all spherical; the seams have a likely effect of the trajectory of the ball. Some pitchers use more pressure on the ball with their middle finger with their index finger right next to it. The ball also does not stay the same shape as it did before contact, affecting the center of mass and likely the cross-sectional area. There are many factors and effects to ponder when studying Coors Field.

6 Appendix: Code

Listing 1. The Code

```
# -*- coding: utf-8 -*-
"""
Created on Sat Nov 21 11:45:53 2020

@author: thoma
"""


import numpy as np
import matplotlib.pyplot as plt

#input game temperature
T = int(input('Input game temperature in Fahrenheit at Coors: '))
print('')

#air density function
def rho(y, T0):
    a = 6.5E-3
    rho0 = 1.079    #average humidity air density
    alpha = 2.7
    return rho0*(1 - (a*y)/(T0))**alpha

#initial values for calculating air density
y = 5200*0.3048     #converting Coors field's elevation to meters
T0 = (T - 32)*5/9 + 273.15
print('Coors Field is {:4.2f} m above sea level'.format(y))
print('The air density at Coors Field is roughly {:4.4f} kg/m^3'.format(rho(y, T0)))


#importing data
hdata = np.loadtxt('kyle_home.txt')
rdata = np.loadtxt('kyle_road.txt')

#home data for average horizontal/vertical break on a fastball per game
xh = hdata[:,0]
yh = hdata[:,1]

#road data for average horizontal/vertical break on a fastball per game
xr = rdata[:,0]
yr = rdata[:,1]

#plotting graph
fig = plt.figure()
plt.title('Horizontal Break (in) vs. Vertical Break (in)')
plt.plot(xh, yh, '.', color = 'purple', label = 'Coors')
plt.plot(xr, yr, '.', color = 'orange', label = 'Road')
plt.xlabel('Horizontal break (in)')
plt.ylabel('Vertical break (in)')
plt.legend()
plt.show()
```

```
print("Freeland's␣average␣horizontal␣break␣of␣a␣fastball␣at␣Coors␣is␣{:4.1f}␣inches".format(np.mean(
print("Freeland's␣average␣vertical␣break␣of␣a␣fastball␣at␣Coors␣is␣{:4.1f}␣inches".format(np.mean(yh
print("Freeland's␣average␣horizontal␣break␣of␣a␣fastball␣on␣the␣road␣is␣{:4.1f}␣inches".format(np.me
print("Freeland's␣average␣vertical␣break␣of␣a␣fastball␣on␣the␣road␣is␣{:4.1f}␣inches".format(np.mean


from mpl_toolkits.mplot3d import Axes3D

#define the drag coefficient function
def k_D(v):
    delta = 5.0
    vd = 35.0
    return 0.0039 + 0.0058/(1 + np.exp((v-vd)/delta))

def k_L(v, omega):
    A = 0.00442      #cross-sectional area of baseball
    m = 0.145        #mass of baseball in kg
    R = 0.075*0.5
    return (omega*R*A)/(2*m*v*60.0*2*np.pi)

#define euler algorithm
def euler(vx,vy,vz, phi):
    #initial conditions
    x = 0
    y = 0
    z = 1.8  #release height in meters
    t = 0
    h = 0.001  #time step
    rho = 0.976      #value calculated from air density function
    rho0 = 1.225     #dry air density
    g = 9.81

    #creating arrays
    global X,Y,Z
    X = np.zeros(0)
    Y = np.zeros(0)
    Z = np.zeros(0)


    while x <= 18.44:  #distance to home base from pitcher's mound
        X = np.append(X, x)
        Y = np.append(Y, y)
        Z = np.append(Z, z)

        v = np.sqrt(vx**2 + vy**2 + vz**2)
        #calculate acceleration components
        ax = -k_D(v)*(rho/rho0)*v*vx + k_L(v, omega)*rho*(vz*omega*np.sin(phi) - vy*omega*np.cos(phi
        ay = -k_D(v)*(rho/rho0)*v*vy + k_L(v, omega)*rho*(vx*omega*np.cos(phi))
        az = -k_D(v)*(rho/rho0)*v*vz - k_L(v, omega)*rho*vx*omega*np.sin(phi) - g

        #apply Euler algorithm
        vx = vx + ax*h
        vy = vy + ay*h
        vz = vz + az*h
        x = x + vx*h
        y = y + vy*h
        z = z + vz*h
        t = t + h

def euler2(vx,vy,vz, phi):
    #initial conditions
    x = 0
    y = 0
    z = 1.8  #release height in meters
    t = 0
    h = 0.001  #time step
    rho = 1.225
    g = 9.81

    global X1,Y1,Z1
    X1 = np.zeros(0)
    Y1 = np.zeros(0)
    Z1 = np.zeros(0)

    while x <= 18.44:  #distance to home base from pitcher's mound
        X1 = np.append(X1, x)
        Y1 = np.append(Y1, y)
        Z1 = np.append(Z1, z)

        v = np.sqrt(vx**2 + vy**2 + vz**2)
        #calculate acceleration components
        ax = -k_D(v)*v*vx + k_L(v, omega)*rho*(vz*omega*np.sin(phi) - vy*omega*np.cos(phi))
        ay = -k_D(v)*v*vy + k_L(v, omega)*rho*(vx*omega*np.cos(phi))
        az = -k_D(v)*v*vz - k_L(v, omega)*rho*vx*omega*np.sin(phi) - g

        #apply Euler algorithm
        vx = vx + ax*h
        vy = vy + ay*h
        vz = vz + az*h
        x = x + vx*h
        y = y + vy*h
        z = z + vz*h
        t = t + h


TYPE = str(input("Type of pitch:  Fastball(f), Curveball(c), Slider(s), Changeup(ch) "))
if TYPE == 'c' or TYPE == 'C':
    v = 35.76   #initial velocity
    phi = 135*np.pi/180.0   #spin axis
    omega = 2444/60.0*2*np.pi   #spin rate
    theta = 1*np.pi/180.0  # angle from horizontal

    #velocities in each direction
    vx = v*np.cos(theta)
    vy = 0*v*np.sin(theta)
    vz = v*np.sin(theta)

    euler(vx,vy,vz, phi)
    euler2(vx,vy,vz,phi)

    #calculating horizontal and vertical break
    print('Horizontal break = {:4.1f} inches'.format((Y1[-1] - Y[-1])*39.37))
    print('Vertical break = {:4.1f} inches'.format((Z1[-1] - Z[-1])*39.37))

if TYPE == 's' or TYPE == 'S':
    v = 38.45
    phi = 180*np.pi/180
    omega = 2473.0/60.0*2*np.pi
    theta = 0*np.pi/180.0  # angle from horizontal

    vx = v*np.cos(theta)
    vy = 0*v*np.sin(theta)
    vz = v*np.sin(theta)
    euler(vx,vy,vz, phi)
    euler2(vx,vy,vz,phi)
    print('Horizontal␣break␣=␣{:4.1f}␣inches'.format((Y1[-1] - Y[-1])*39.37))
    print('Vertical␣break␣=␣{:4.1f}␣inches'.format((Z1[-1] - Z[-1])*39.37))

if TYPE =='f' or TYPE == 'F':
    v = 41.13
    phi = 315.0*np.pi/180
    omega = 2451/60.0*2*np.pi
    theta = -0*np.pi/180.0  # angle from horizontal

    vx = v*np.cos(theta)
    vy = 0*v*np.sin(theta)
    vz = v*np.sin(theta)

    euler(vx,vy,vz, phi)
    euler2(vx,vy,vz,phi)
    print('Horizontal␣break␣=␣{:4.1f}␣inches'.format((Y1[-1] - Y[-1])*39.37))
    print('Vertical␣break␣=␣{:4.1f}␣inches'.format((Z1[-1] - Z[-1])*39.37))

if TYPE == 'ch' or TYPE == 'CH':
    v = 38.45
    phi = 0*np.pi/180
    omega = 1412.0/60.0*2*np.pi
    theta = -0*np.pi/180.0  # angle from horizontal

    vx = v*np.cos(theta)
    vy = 0*v*np.sin(theta)
    vz = v*np.sin(theta)

    euler(vx,vy,vz, phi)
    euler2(vx,vy,vz,phi)
    print('Horizontal␣break␣=␣{:4.1f}␣inches'.format((Y1[-1] - Y[-1])*39.37))
    print('Vertical␣break␣=␣{:4.1f}␣inches'.format((Z1[-1] - Z[-1])*39.37))

#plotting simulation
fig = plt.figure()
ax = Axes3D(fig)
ax.set_xlabel('X␣axis')
ax.set_ylabel('Y␣axis')
ax.set_zlabel('Z␣axis')
ax.plot(X,Y, zs = Z, zdir = 'z', color = 'purple', label = 'Coors')
ax.plot(X1, Y1, zs = Z1, zdir = 'z', color = 'orange', label = 'Other')
plt.show()

print('')
from scipy import stats

homedata = np.loadtxt('nolan␣home.txt')
roaddata = np.loadtxt('nolan␣road.txt')

#home run distance, exit velocity, and launch angle respectively
Hdis = homedata[:,0]
Hev = homedata[:,1]
Hla = homedata[:,2]

Rdis = roaddata[:,0]
Rev = roaddata[:,1]
Rla = roaddata[:,2]
print("Arenado's␣average␣home␣run␣distance␣at␣Coors␣is␣{:4.1f}␣ft".format(np.mean(Hdis)))
print("Arenado's␣average␣home␣run␣distance␣at␣other␣ballparks␣is␣{:4.1f}␣ft".format(np.mean(Rdis)))
print("Arenado's␣average␣exit␣velocity␣at␣Coors␣is␣{:4.1f}␣mph".format(np.mean(Hev)))
print("Arenado's average exit velocity at other ballparks is {:4.1f} mph".format(np.mean(Rev)))
print("Arenado's average launch angle at Coors is {:4.1f} degrees".format(np.mean(Hla)))
print("Arenado's average launch angle at other ballparks is {:4.1f} degrees".format(np.mean(Rla)))
print('Arenado has hit {:4.0f} home runs at Coors'.format(len(Hdis)))
print('Arenado has hit {:4.0f} home runs at other ballparks'.format(len(Rdis)))

#linear regression
slope, intercept, r_value, p_value, std_err = stats.linregress(Hev, Hdis)
A = intercept
B = slope
x = np.linspace(min(Hev), max(Hev))
y = B*x + A
slope2, intercept2, r_value2, p_value2, std_err2 = stats.linregress(Rev, Rdis)
D = intercept2
C = slope2
x2 = np.linspace(min(Rev), max(Rev))
y2 = C*x2 + D

print('')
print('Home slope = ', B)
print('Home intercept = ', A)
print('Road slope =', C)
print('Road intercept =', D)
print('')

#plotting data and linear regression
fig = plt.figure()
plt.title('Nolan Arenado: Exit Velocity vs. HR Distance')
plt.plot(Hev, Hdis, '.', color = 'purple', label = 'Home Runs at Coors')
plt.plot(Rev, Rdis, '.', color = 'orange', label = 'Home Runs on the road')
plt.plot(x, y, color = 'purple')
plt.plot(x2, y2, color = 'orange')
plt.xlabel('Exit velocity (mph)')
plt.ylabel('HR Distance (ft)')
plt.legend()

#calculating difference between means of home and away distance
diff = np.mean(Hdis) - np.mean(Rdis)
print('There is a {:4.2f} ft difference between the average distance of a home run hit at Coors and

#new euler functions for home runs
def eulerhr(vx,vy,vz, phi):
    #initial conditions
    x = 0
    y = 0
    z = 0.6  #release height in meters
    t = 0
    h = 0.001  #time step
    rho = 0.976
    rho0 = 1.225
#    omega = 1800.0/60.0*2*pi
    g = 9.81
    global X,Y,Z
    X = np.zeros(0)
    Y = np.zeros(0)
    Z = np.zeros(0)


    while z>= 0:  #distance to home base from pitcher's mound
        X = np.append(X, x)
        Y = np.append(Y, y)
        Z = np.append(Z, z)
        v = np.sqrt(vx**2 + vy**2 + vz**2)
        #calculate acceleration components
        ax = -k_D(v)*(rho/rho0)*v*vx + k_L(v, omega)*rho*(vz*omega*np.sin(phi) - vy*omega*np.cos(phi
        ay = -k_D(v)*(rho/rho0)*v*vy + k_L(v, omega)*rho*(vx*omega*np.cos(phi))
        az = -k_D(v)*(rho/rho0)*v*vz - k_L(v, omega)*rho*vx*omega*np.sin(phi) - g

        #apply Euler algorithm
        vx = vx + ax*h
        vy = vy + ay*h
        vz = vz + az*h
        x = x + vx*h
        y = y + vy*h
        z = z + vz*h
        t = t + h

def eulerhr2(vx,vy,vz, phi):
    #initial conditions
    x = 0
    y = 0
    z = 0.6  #release height in meters
    t = 0
    h = 0.001  #time step
    rho = 1.225

#    omega = 1800.0/60.0*2*pi
    g = 9.81
    global X1,Y1,Z1
    X1 = np.zeros(0)
    Y1 = np.zeros(0)
    Z1 = np.zeros(0)


    while z >= 0:  #distance to home base from pitcher's mound
        X1 = np.append(X1, x)
        Y1 = np.append(Y1, y)
        Z1 = np.append(Z1, z)

        v = np.sqrt(vx**2 + vy**2 + vz**2)
        #calculate acceleration components
        ax = -k_D(v)*v*vx + k_L(v, omega)*rho*(vz*omega*np.sin(phi) - vy*omega*np.cos(phi))
        ay = -k_D(v)*v*vy + k_L(v, omega)*rho*(vx*omega*np.cos(phi))
        az = -k_D(v)*v*vz - k_L(v, omega)*rho*vx*omega*np.sin(phi) - g

        #apply Euler algorithm
        vx = vx + ax*h
        vy = vy + ay*h
        vz = vz + az*h
        x = x + vx*h
        y = y + vy*h
        z = z + vz*h
        t = t + h
v = 45.662564530973455135
phi = 270.0*np.pi/180
omega = 2337/60.0*2*np.pi

theta = 29*np.pi/180.0  # angle from horizontal

vx = v*np.cos(theta)
vy = v*np.sin(theta)
vz = v*np.sin(theta)
```

```
eulerhr(vx,vy,vz, phi)
eulerhr2(vx,vy,vz, phi)

#calculating distance traveled of ball according to euler function
Hdis = np.sqrt(Y[-1]**2 + X[-1]**2)*3.28084
Rdis = np.sqrt(Y1[-1]**2 + X1[-1]**2)*3.28084


print('')
print('Average home run distance at Coors is {:4.4} ft'.format(Hdis))
print('Average home run distance at other ballparks is {:4.4} ft'.format(Rdis))
print('The added distance at Coors is {:4.4} ft'.format(Hdis - Rdis))

fig = plt.figure()
ax = Axes3D(fig)
ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')
ax.set_xlim3d(0,120)
ax.set_ylim3d(0,120)
ax.set_zlim3d(0,120)
ax.plot(X,Y, zs = Z, zdir = 'z', color = 'purple', label = 'Coors')
ax.plot(X1, Y1, zs = Z1, zdir = 'z', color = 'orange', label = 'Other')
plt.show()
```