ME 459/5559 – Robotics and Unmanned Systems
HW #7: DUE May 2nd, 2022

The focus of this assignment is on the development of modeling and sensitivity analysis through Monte Carlo. The code developed and shown during lecture is provided on Canvas as one way to implement Monte Carlo. Additionally, a parachute simulation is posted on canvas in an attempt to assist you in building your framework code.

Problem 1:

You are tasked with creating a Monte Carlo simulation of a parachute/payload that is deployed from an aircraft. Use the parachute simulation posted on Canvas to help get you started. This simulation is 3 degrees-of-freedom (X, Y, and Z). The simulation posted simply takes in the simulation parameters and runs a single simulation. Note, that the parachute size changes from a drogue (small) to a main (large) parachute at the z_transition altitude. The transition altitude can have a major impact on the landing location (large parachute floats in wind longer).

Your task is to create a Monte Carlo simulation (at least 100 simulations per transition altitude) that looks at the sensitivity of the transition altitude on the landing location (both the actual location and the spread of the location).

Conduct 5 Monte Carlo simulations with transition altitudes of [5000, 4000, 3000, 2000, 1750].

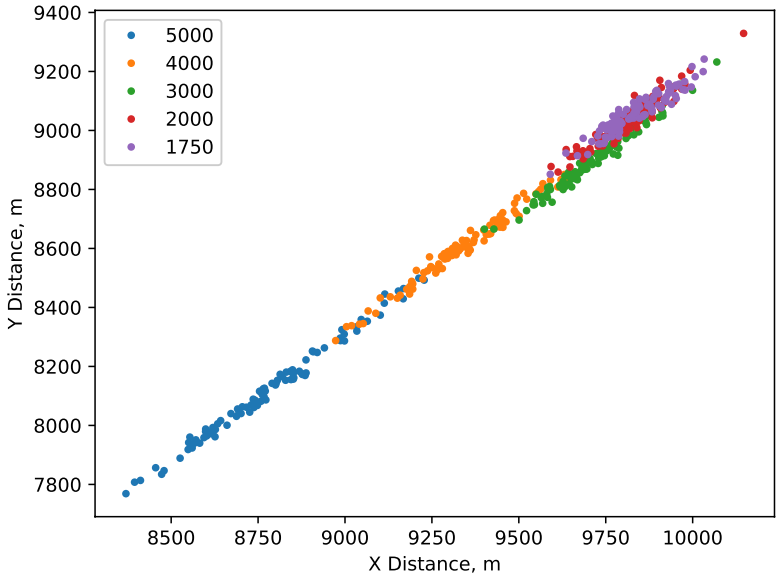
Use the following randomization parameters in your Monte Carlo simulation.

Parameter	Mean Value	Standard Deviation	Distribution
C_D	0.7	0.1	Normal
Wind Vel	0.1 m/s	N/A	Chi^2 (k=2)
Runin Angle	45°	5°	Normal
Runin Speed	75 m/s	4 m/s	Normal

For wind speed, see the example in the cannon ball Monte Carlo simulation.

The desired landing location (known is impact point [IP]) is **(3000m, 2000m)**, with a tolerance radius of **100m**. For each transition altitude, show the average distance from the IP, standard deviation, and percentage of simulations that are within the tolerance. An example is shown below for reference (made up numbers). Provide a plot showing the landing locations (different colors for each transition altitude). An example is shown below for reference.

Transition Alt	Mean Distance from Target	StD Distance	% in Bounds
5000	270	20	0%
4000	300	10	0%
3000	473	16	71%
2000	427	42	15%
1750	140	74	0%



Problem 2:

In this problem, you need to use the same Monte Carlo environment you created for Problem 1 with a few tweaks. We now have a mandatory No-Go zone (this could be a cliff, bad guy territory, etc.). The payload can not land in this area. **You are tasked with selecting the best run in angle that will still on average land close to the target, but does not enter the no-go zone (at a reasonably high confidence).**

Simulation Parameters:

Parameter	Mean Value	Standard Deviation	Distribution
Z_{start}	5000m	-	-
Z_{end}	1500m	-	-
$Z_{transition}$	3000m	75m	Normal
C_D	0.7	0.1	Normal
Wind Vel	0.1 m/s	N/A	Chi² (k=2)
Runin Speed	75 m/s	6 m/s	Normal

Run in angle search space: 10 distinct angles between 0° and 360°. Use np.linspace(0.0,6.28,11) to create 10 distinct angles (0 and 6.28 are the same angle). Use 100 simulations for each run in angle.

Keepout zone is a rectangle with a bottom left and top right coordinate of (2030m, 1110m) and (2130m, 1200m), respectively. The desired landing location is (2005m, 1088m).

A simple check function is helpful in storing the number of simulations that are outside of the no-go zone.

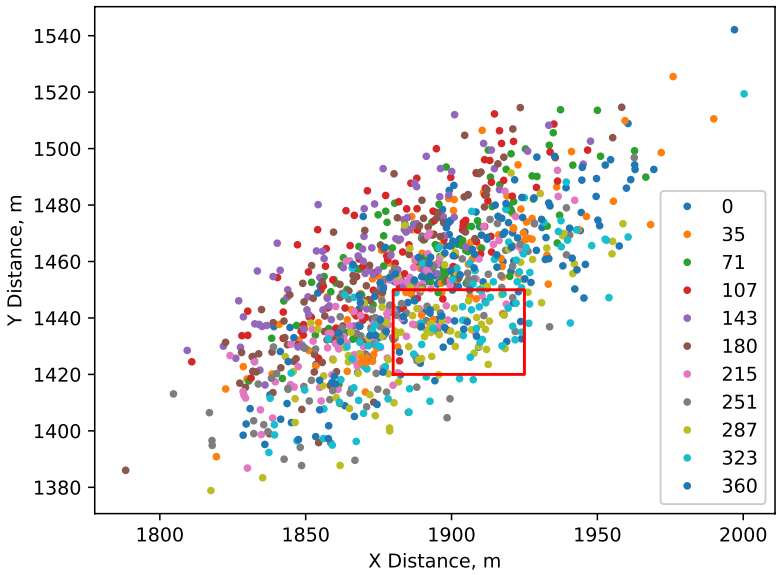
```
# Function to see point X,Y is inside of a rectangle
def box_check(x1, y1, x2, y2, x, y):
    if (x > x1 and x < x2 and y > y1 and y < y2):
        return False # If inside box, return FALSE
    else:
        return True
```

When beginning, start with only a few run in angles and simulations to make sure your code is working properly. The simulations take a bit to run.

Provide a table similar to Problem #1 (example shown below), and a plot that shows all landing locations accompanied with the No-Go zone and the desired landing location. The code below may help with drawing the rectangle from two points:

plt.plot([keepout_x[0],keepout_x[1], keepout_x[1], keepout_x[0], keepout_x[0]], [keepout_y[0], keepout_y[0], keepout_y[1], keepout_y[1], keepout_y[0]], 'r-')

Run in Angle	Mean Distance from Target	StD Distance from Target	% Outside No-Go Zone
0°	51	30	83%
36	52	30	89%
72	49	24	98%
108	45	21	96%
144	41	24	98%
180	30	20	99%
216	35	22	88%
252	36	23	84%
288	44	27	61%
324	48	27	74%



Make sure to discuss which run in angle you think is the best for this scenario. Comment (briefly) on your answer.