

Diagnostic 11

Modeling and Simulation
Fall 2017

This diagnostic covers material from Chapter 11 of *Modeling and Simulation in Python*.

Vocabulary

The rate of change in angular velocity is called _____ and typically denoted with the Greek letter _____.

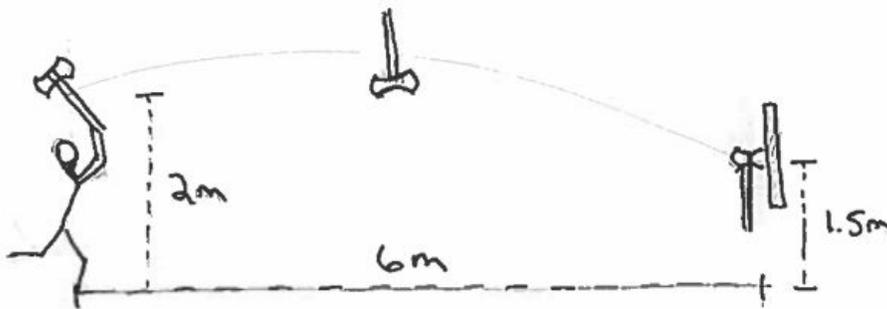
Given a force and a moment arm in the form of vectors, we can compute torque using the _____ operation.

Mass is what makes an object hard to accelerate; _____ is what makes an object hard to rotate.

In SymPy, the function that solves systems of equations is _____.

Code

Our favorite event at Lumberjack Competitions is axe throwing. The axes used for this event typically weigh 1.5 to 2 kg, with handles roughly 0.7 m long. They are thrown overhead at a target typically 6 m away and 1.5 m off the ground. Normally, the axe makes one full rotation in the air to hit the target blade first, with the handle close to vertical.



Assume that we have written the following `make_system` function:

```

def make_system():
    P = Vector(0, 2)    # position in m
    V = Vector(8, 4)    # velocity in m/s

    duration = 1.0
    init = State(x=P.x, y=P.y, theta=2,
                 vx=V.x, vy=V.y, omega=-7)
    ts = linspace(0, duration, 101)

    return System(init=init, ts=ts,
                  g = 9.8,          # acceleration of gravity
                  mass = 1.5,       # mass in kg
                  length = 0.7)     # length in m

```

1) Write a slope function for this system (called `slope_func`, of course). As a simple starting place, you can ignore drag, so the only force on the axe is gravity.

2) Write a line of code to call `odeint` with this system and slope function.

3) Write a few lines of code to extract the x and y positions of the axe's center of gravity as a function of time, and plot the results as a trajectory.

If you'd like to see this simulation in action, pull from upstream, then run `throwingaxe.ipynb`. If you are so inclined, work on the exercises you find there (or look at `throwingaxesoln.ipynb`).