```
 1: import numpy as np
 2: import pylab
 3: from multiprocessing import Pool, cpu_count
 4:
 5:
 6: def RungeKutta(func, init, a, b, h=0.1):
 7:     X = []
 8:     T = np.arange(a, b, h)
 9:     x = init
10:     for t in T:
11:         X.append(x)
12:         k1 = h * func(x, t)
13:         k2 = h * func(x + k1 / 2, t + h / 2)
14:         k3 = h * func(x + k2 / 2, t + h / 2)
15:         k4 = h * func(x + k3, t + h)
16:         x += (1 / 6) * (k1 + 2 * k2 + 2 * k3 + k4)
17:     return T, X
18:
19:
20: def RungeKutta2(f1, f2, x_init, y_init, a, b, h=0.1):
21:     X = []
22:     Y = []
23:     T = np.arange(a, b, h)
24:     x = x_init
25:     y = y_init
26:     for t in T:
27:         if(x > np.pi):
28:             x -= 2 * np.pi
29:         if(x < -np.pi):
30:             x += 2 * np.pi
31:         if (2*t/3) % (2*np.pi) <= h and t > 100:
32:             X.append(x)
33:             Y.append(y)
34:         k1 = h * f1(x, y, t)
35:         l1 = h * f2(x, y, t)
36:         k2 = h * f1(x + k1 / 2, y + l1 / 2, t + h / 2)
37:         l2 = h * f2(x + k1 / 2, y + l1 / 2, t + h / 2)
38:         k3 = h * f1(x + k2 / 2, y + l2 / 2, t + h / 2)
39:         l3 = h * f2(x + k2 / 2, y + l2 / 2, t + h / 2)
40:         k4 = h * f1(x + k3, y + l3, t + h)
41:         l4 = h * f2(x + k3, y + l3, t + h)
42:         x += (1 / 6) * (k1 + 2 * k2 + 2 * k3 + k4)
43:         y += (1 / 6) * (l1 + 2 * l2 + 2 * l3 + l4)
44:     return T, X, Y
45:
46: def helper(fD):
47:     g = 9.8
48:     l = 9.8
49:     q = 0.5
50:     t_max = 200
51:     OmegaD=2/3
52:     f1 = lambda theta,omega,t: omega
53:     f2 = lambda theta,omega,t: -(g/l)*np.sin(theta)-q*omega+fD*np.sin(OmegaD*t)
54:     T, Theta, Omega = RungeKutta2(f1, f2, 0.2, 0, 0, t_max, 0.001)
55:     return [(fD, x) for x in Theta]
56:
57:
58: def main():
59:     FD = np.linspace(1.35, 1.6, 50)
60:     pool = Pool(4)
61:     pts  = pool.map(helper, FD)
62:     pts = [j for i in pts for j in i]
63:     pylab.plot(*zip(*pts), 'k.')
```

```
64:
65:     pylab.show()
66:
67:
68: if __name__ == "__main__":
69:     main()
```