shootingRootFindExample

December 6, 2022

Let us start with a very simple problem of freefall of a ball under the effect of gravity. from https://www.earthinversion.com/techniques/solving-boundary-value-problems-using-the-shooting-method/

d 2 → y d t 2 = − g

(7)

where, g = 9.8 m/s^2. The boundary condition here is that we start at y = 0 with unknown velocity, but we know that it will be back at t = 10 .

[2]:
```
import numpy as np
from scipy.integrate import solve_ivp
from scipy import optimize
import matplotlib.pyplot as plt
plt.style.use('seaborn')

g = 9.8
y0 = 0
tf = 10



def f(t, r):
    y, v = r
    dy_dt = v  # velocity
    d2y_dt2 = -g  # acceleration

    return dy_dt, d2y_dt2

@np.vectorize
def shooting_eval(v0):
    sol = solve_ivp(f, (t[0], t[-1]), (y0, v0), t_eval=t)
    y_num, v = sol.y
    return y_num[-1]


v0 = 60  # guess the initial velocity
t = np.linspace(0, tf, 51)
```

```
fig, ax = plt.subplots()
v0 = np.linspace(0, 100, 100)
plt.plot(v0, shooting_eval(v0), label='Shooting evaluation')
plt.axhline(c="k")

# root finding: secant method (kind of Newton Raphson method where slope is␣
 ↪unknown)
# Find a zero of the function func given a nearby starting point x0
v0 = optimize.newton(shooting_eval, 50)
print(v0)
plt.plot(v0, 0, 'ro', label=f'Solution: {v0:.1f}')

plt.grid(True)
plt.legend()
ax.set_xlabel('t')
ax.set_ylabel('y')
plt.savefig('example1_solution.png', bbox_inches='tight', dpi=300)
plt.close()
```

```
48.999999999999964
```

[3]:

```
# Plot the path

t = np. linspace(0, tf, 51)
sol = solve_ivp(f, (0, tf), (y0, v0), t_eval=t)
y, v = sol.y

plt.plot(t[0], y0, 'ro', label=f'Solution: {v0:.1f}')
plt.plot(t[-1], 0, 'ro', label=f'Solution: {v0:.1f}')
plt.plot(t, y, ".")

plt.savefig('example1_solution_path.png', bbox_inches='tight', dpi=300)
plt.close()
```

Example 2 Let us take another simple differential equation to solve.

d 2 $\rightarrow$ y d t 2 + 3 y = 0

(8)

where, y (0) = 7 and y (2 $*$) = 0 .

[5]:

```
import numpy as np
from scipy.integrate import solve_ivp
from scipy import optimize
import matplotlib.pyplot as plt
plt.style.use('seaborn')
```

```python
# example d2y/dt2+3y = 0
# y(0) = 7 and y(2*pi)=0

y0 = 7
tf = 2 * np.pi


def f(t, r):
    y, v = r
    dy_dt = v  # velocity
    d2y_dt2 = -3 * y  # acceleration

    return dy_dt, d2y_dt2


@np.vectorize
def shooting_eval(v0):
    sol = solve_ivp(f, (t[0], t[-1]), (y0, v0), t_eval=t)
    y_num, v = sol.y
    return y_num[-1]


v0 = 60  # guess the initial velocity
t = np.linspace(0, 2 * np.pi, 100)

fig, ax = plt.subplots()
v0 = np.linspace(0, 100, 100)
plt.plot(v0, shooting_eval(v0), label='Shooting evaluation')
plt.axhline(c="k")


# root finding: secant method (kind of Newton Raphson method where slope is␣
↪unknown)
# Find a zero of the function func given a nearby starting point x0
v0 = optimize.newton(shooting_eval, 50)
print(v0)
plt.plot(v0, 0, 'ro', label=f'Solution: {v0:.1f}')


plt.grid(True)
plt.legend()
ax.set_xlabel('t')
ax.set_ylabel('y')
plt.savefig('example2_solution.png', bbox_inches='tight', dpi=300)
plt.close()
```

```
-1.3366717449748615
```

[6]:

```
# Plot the path

t = np. linspace(0, tf, 51)
sol = solve_ivp(f, (0, tf), (y0, v0), t_eval=t)
y, v = sol.y

plt.plot(t[0], y0, 'ro', label=f'Solution: {v0:.1f}')
plt.plot(t[-1], 0, 'ro', label=f'Solution: {v0:.1f}')
plt.plot(t, y, ".")

plt.savefig('example2_solution_path.png', bbox_inches='tight', dpi=300)
plt.close()
```

[ ]:

```

```