

```
1: import week6
2: import sgd
3: import pandas as pd
4: import matplotlib.pyplot as plt
5: import numpy as np
6: import sys
7:
8: def runp(alpha=0.5, beta=0.9, n=5):
9:     T = pd.read_csv("data/T.csv").values
10:    fg = week6.generate_optimisation_functions(
11:        T, minibatch_size=n, seed=None)
12:    o = sgd.StochasticGradientDescent()
13:    start = np.array([3, 3])
14:    o.start(start)
15:    o.step_size(alpha)
16:    o.beta(beta)
17:    o.alg("rmsprop")
18:    o.function_generator(fg)
19:    xs = []
20:    fs = []
21:    xs.append(o._x_value)
22:    fs.append(week6.f(o._x_value, T))
23:    for i in range(200):
24:        o.step()
25:        xs.append(o._x_value)
26:        fs.append(week6.f(o._x_value, T))
27:    return {
28:        "x1": [x[0] for x in xs],
29:        "x2": [x[1] for x in xs],
30:        "f": fs,
31:    }
32:
33:
34:
35: x_min, x_max, y_min, y_max = [-5, 5, -5, 5]
36: T = pd.read_csv("data/T.csv").values
37: # Generate data for wireframe plot
38: resolution = 100
39: x_range = np.linspace(x_min, x_max, resolution)
40: y_range = np.linspace(y_min, y_max, resolution)
41: X, Y = np.meshgrid(x_range, y_range)
42:
43: # Plot wireframe
44: fig = plt.figure(figsize=(12, 6))
45: resolution = 100
46: Z_contour = np.zeros_like(X)
47: for i in range(resolution):
48:     for j in range(resolution):
49:         Z_contour[i, j] = week6.f([X[i, j], Y[i, j]], T)
50:
51: # Plot contour
52: ax_contour = fig.add_subplot(122)
53: contour = ax_contour.contourf(X, Y, Z_contour, levels=20, cmap='viridis')
54: plt.colorbar(contour, ax=ax_contour, label='$f_T(x)$')
55: ax_contour.set_xlabel('$x_1$')
56: ax_contour.set_ylabel('$x_2$')
57: ax_contour.set_xlim([-5, 5])
58: ax_contour.set_ylim([-5, 5])
59: plt.suptitle('Stochastic Gradient Descent with RMSProp step on $f_T(x)$')
60:
61: ax_f = fig.add_subplot(121)
62:
63: alpha = 0.1
64: beta = 0.99
65: for n in [1, 3, 5, 7, 15, 25]:
66:     np.random.seed(57)
67:     T = pd.read_csv("data/T.csv").values
68:     run = runp(n=n)
69:     label = f"$n={n}$, $\alpha={alpha}$, $\beta={beta}$, best $f_T(x)={min(run['f']):.3f}$"
70:     ax_contour.plot(run["x1"], run["x2"], label=label)
71:     ax_f.plot(run['f'], label=label)
72:
73: ax_f.set_yscale('log')
74: ax_f.set_xlabel("iteration $t$")
75: ax_f.set_ylabel("$f_T(x_t)$")
76: ax_f.legend(loc="upper right")
77: plt.savefig("fig/cii-2.pdf")
78: plt.show()
```