```python
 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, beta2=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.beta2(beta)
18:     o.alg("adam")
19:     o.function_generator(fg)
20:     xs = []
21:     fs = []
22:     xs.append(o._x_value)
23:     fs.append(week6.f(o._x_value, T))
24:     for i in range(200):
25:         o.step()
26:         xs.append(o._x_value)
27:         fs.append(week6.f(o._x_value, T))
28:     return {
29:         "x1": [x[0] for x in xs],
30:         "x2": [x[1] for x in xs],
31:         "f": fs,
32:     }
33:
34:
35:
36: x_min, x_max, y_min, y_max = [-5, 5, -5, 5]
37: T = pd.read_csv("data/T.csv").values
38: # Generate data for wireframe plot
39: resolution = 100
40: x_range = np.linspace(x_min, x_max, resolution)
41: y_range = np.linspace(y_min, y_max, resolution)
42: X, Y = np.meshgrid(x_range, y_range)
43:
44: # Plot wireframe
45: fig = plt.figure(figsize=(12, 6))
46: resolution = 100
47: Z_contour = np.zeros_like(X)
48: for i in range(resolution):
49:     for j in range(resolution):
50:         Z_contour[i, j] = week6.f([X[i, j], Y[i, j]], T)
51:
52: # Plot contour
53: ax_contour = fig.add_subplot(122)
54: contour = ax_contour.contourf(X, Y, Z_contour, levels=20, cmap='viridis')
55: plt.colorbar(contour, ax=ax_contour, label='$f_T(x)$')
56: ax_contour.set_xlabel('$x_1$')
57: ax_contour.set_ylabel('$x_2$')
58: ax_contour.set_xlim([-5, 5])
59: ax_contour.set_ylim([-5, 5])
60: plt.suptitle('Gradient Descent with Adam step on $f_T(x)$')
61:
62: ax_f = fig.add_subplot(121)
63:
64: np.random.seed(57)
65: T = pd.read_csv("data/T.csv").values
66: for alpha in [1, 2, 3, 4, 5, 10]:
67:     for beta in [0.9]:
68:         for beta2 in [0.9]:
69:             alpha_a = alpha # * (1-beta)
70:             n = 25
71:             run = runp(n=n, alpha=alpha, beta=beta, beta2=beta2)
72:             label = f"$\\alpha={alpha_a}$, $\\beta_1={beta}$, $\\beta_2={beta2}$"
73:             ax_contour.plot(run["x1"], run["x2"], label=label)
74:             ax_f.plot(run['f'], label=label)
75:
76: ax_f.set_yscale('log')
77: ax_f.set_xlabel("iteration $t$")
78: ax_f.set_ylabel("$f_T(x_t)$")
79: ax_f.legend(loc="upper right")
80: plt.savefig("fig/civ-2.pdf")
81: plt.show()
```