```python
import global_random_search
import lib
import numpy as np
import sgd
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
import pandas as pd

f = {
    "function": lib.f_real,
    "gradient": lib.f_grad,
}

g = {
    "function": lib.g_real,
    "gradient": lib.g_grad,
}


def gradient_descent_constant(step_size=0.0065, start=[0, 0], funcs=f, max_iter=10000, exp="exp/aii-gd-constant.csv"):
    start = np.array(start)
    g = sgd.StochasticGradientDescent()
    g.max_iter(max_iter)
    g.step_size(step_size)
    g.start(start)
    def function_generator():
        while True:
            yield funcs["function"], funcs["gradient"]
    g.function_generator(function_generator())
    g.debug(True)
    g.alg("constant")
    for i in range(max_iter):
        g.step()
        yield {
                "x": g._x_value,
        }

if __name__ == "__main__":
    res = list(gradient_descent_constant(max_iter=1000))
    res = pd.DataFrame(res)
    res["f(x)"] = res["x"].apply(f["function"])
    print(res)

    ps = [{"min": 0, "max": 10}, {"min": 0, "max": 18}]
    grs = global_random_search.a(costf=f["function"], parameters=ps, N=1000)

    plt.figure()

    print(res["f(x)"], len(res["f(x)"]))
    plt.plot(list(range(len(res["f(x)"]))), res["f(x)"], label="gradient descent", color="black")
    costs = grs['stats']['it_best_costs']
    plt.plot(list(range(len(costs))), costs, label="global random search", color="orange")
    plt.title("Global Random Search vs Gradient Descent on $f(x)$")
    custom_lines = [
            Line2D([0], [0], color='black', lw=2),
            Line2D([0], [0], color='orange', lw=2),
            ]
    custom_labels = ['gradient descent', 'rnd search a' ]
    plt.legend(custom_lines, custom_labels)
    plt.yscale('log')
    plt.xlabel('iteration')
    plt.tight_layout()
    plt.savefig('fig/aii-iterations-f.pdf')

    res = list(gradient_descent_constant(max_iter=1000, step_size=0.003, funcs=g))
    res = pd.DataFrame(res)
    res["f(x)"] = res["x"].apply(g["function"])
    print(res)

    ps = [{"min": 0, "max": 10}, {"min": 0, "max": 18}]
    grs = global_random_search.a(costf=g["function"], parameters=ps, N=1000)

    plt.figure()

    print(res["f(x)"], len(res["f(x)"]))
    plt.plot(list(range(len(res["f(x)"]))), res["f(x)"], label="gradient descent", color="black")
    costs = grs['stats']['it_best_costs']
    plt.plot(list(range(len(costs))), costs, label="global random search", color="orange")
    plt.title("Global Random Search vs Gradient Descent on $g(x)$")
    plt.legend()
    plt.yscale('log')
    plt.xlabel('iteration')
    plt.tight_layout()
    plt.savefig('fig/aii-iterations-g.pdf')
```