1.2

March 24, 2024

[1]:
```
import random

import matplotlib.pyplot as plt
import jax
from jax import numpy as np
from jax import scipy as sp
```

[2]:
```
def seed():
    return jax.random.PRNGKey(random.randint(0, 228))
```

[3]:
```
eps = 10**-5
data = {10: [], 100: [], 1000: []}
for n in [10, 100, 1000]:
    for _ in range(100):
        x = np.ones(n)
        diag = []
        kappa = jax.random.uniform(seed()) * 4 + 1
        posmax = random.randint(0, n - 1)
        while True:
            posmin = random.randint(0, n)
            if posmin != posmax:
                break
        for i in range(n):
            if i == posmax:
                diag += [kappa]
                continue
            if i == posmin:
                diag += [1.]
                continue
            diag += [jax.random.uniform(seed()) * (kappa - 1) + 1]
        A = np.diag(np.array(diag))
        def f(x):
            return x.T @ A @ x
        alpha = 2 / (1. + kappa)
        norm = np.linalg.norm(jax.grad(f)(x), ord=2)
        iters = 0
        while np.linalg.norm(jax.grad(f)(x), ord=2)**2 > eps * norm**2:
            x = x - alpha * jax.grad(f)(x)
```

```
            iters += 1
        data[n] += [(kappa, iters)]
```

[4]:
```
plt.figure(figsize=(10, 8))
plt.xlabel('$\kappa$')
plt.ylabel('Number of iterations')

plt.scatter(np.array(data[10])[:, 0], np.array(data[10])[:, 1], marker='*',␣
↪c='b', label='$n = 10$')
plt.scatter(np.array(data[100])[:, 0], np.array(data[100])[:, 1], marker='^',␣
↪c='g', label='$n = 100$')
plt.scatter(np.array(data[1000])[:, 0], np.array(data[1000])[:, 1], marker='+',␣
↪c='r', label='$n = 1000$')

plt.legend()
```

[4]:
```
<matplotlib.legend.Legend at 0x7e85f6d8b7d0>
```