# Homework2 - Problem4 (Sakshi)

February 16, 2022

## 1 Problem 4

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
# function to learn
def g(x, beta):
    return beta[0][0] + np.sin(beta[1][0] * x) + np.cos(beta[2][0] * x)
```

[3]:
```
# calculate reduced sum of squares loss
def RSS(g, x, y, beta):
    return np.sum((g(x, beta) - y)**2)
```

[4]:
```
# calculate gradient
def get_gradient(g, x, y, beta):
    del_g = np.hstack((np.array([x * np.cos(beta[1][0] * x), -x * np.
 ↪sin(beta[2][0] * x)]).reshape((10, 2)), np.ones(x.shape[0]).reshape(-1, 1)))
    gradient = np.sum(2 * (g(x, beta) - y) * del_g, axis=0).reshape((-1, 1))
    return gradient
```

[5]:
```
# implementation of batch gradient descent
def gradient_descent(g, x, y, learning_rate=0.01, iterations=100,␣
 ↪error_threshold=0.001):
    # random parameter initialization
    param = np.array([np.random.rand(), np.random.rand(), np.random.rand()]).
 ↪reshape((-1, 1))
    prev = param

    losses = []
    for i in range(iterations):
        gradient = get_gradient(g, x, y, prev)
        param = prev - learning_rate * gradient
        if np.all(np.abs(param - prev) < error_threshold):
            print(f"Convergence found at iteration {i}.")
            break
        prev = param
        # collect loss w.r.t. number of iterations
```

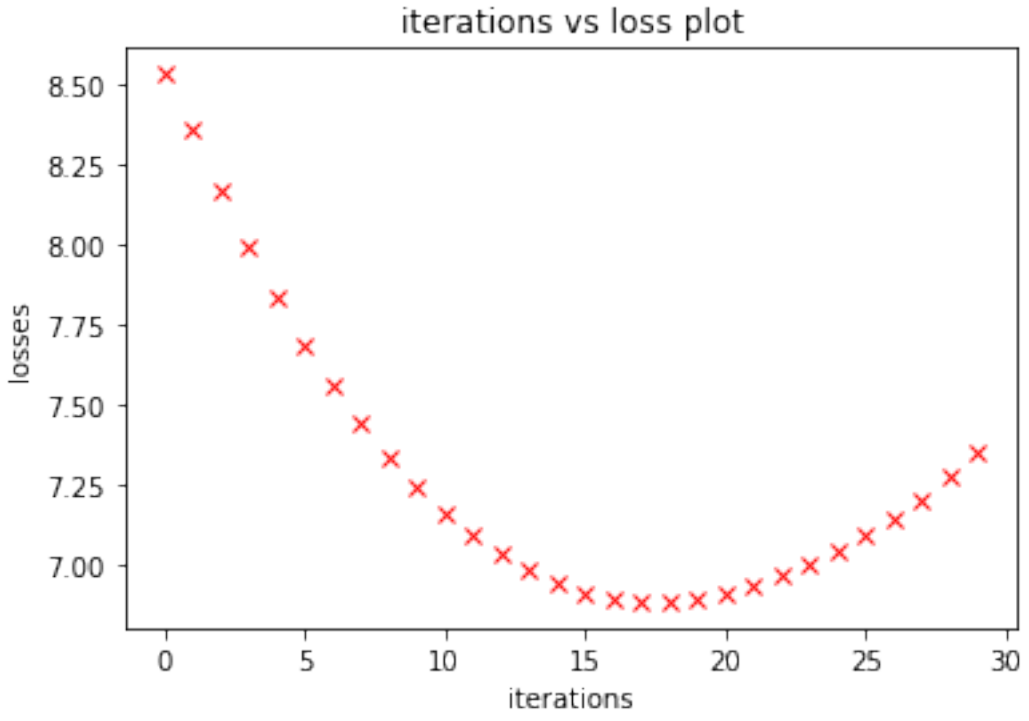
```
        losses.append(RSS(g, x, y, param))

    return losses, param
```

[6]:
```
# input data and labels
x = np.array([0, 2, 4, 6, 8, 10, 12, 14, 16, 18]).reshape((-1, 1))
y = np.array([2.85, 1.5, 0.49, 1.57, 1.9, 0.6, 0.38, 2.33, 1.65, 0.3]).
 ↪reshape((-1, 1))
```

[7]:
```
losses, param = gradient_descent(g, x, y, learning_rate=0.001, iterations=30,␣
 ↪error_threshold=0.0001)
```

[8]:
```
plt.plot(losses, 'rx')
plt.xlabel('iterations')
plt.ylabel('losses')
plt.title('iterations vs loss plot')
plt.show()
```




[ ]:
```

```