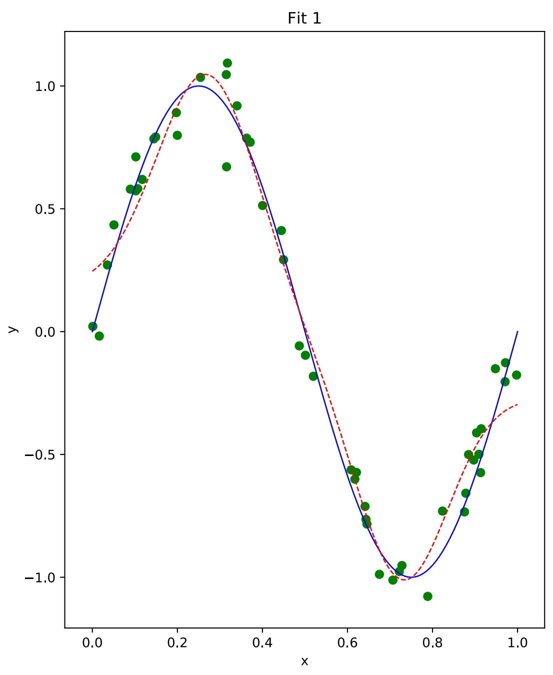
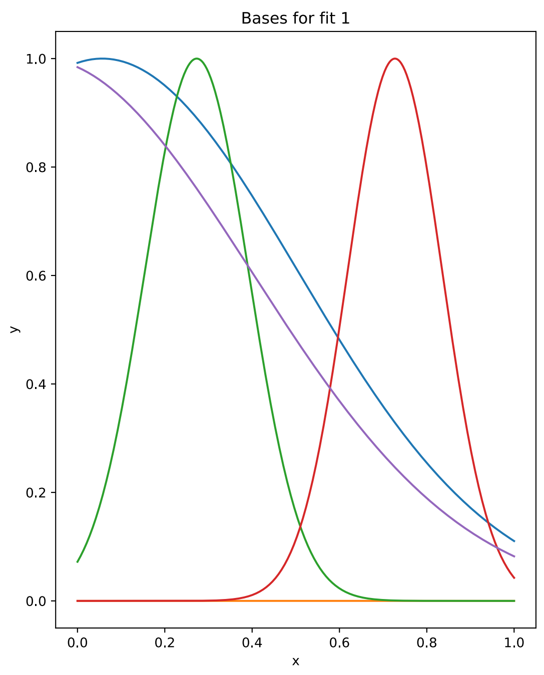
```
#!/usr/local/bin/python3.6

import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt

from tqdm import tqdm

NUM_FEATURES = 5
BATCH_SIZE = 1
NUM_BATCHES = 100

class Data():
    def __init__(self):
        num_samp = 50
        sigma = 0.1
        np.random.seed(31415)

        # We're going to learn these paramters

        self.index = np.arange(num_samp)
        self.x = np.random.uniform(size=(num_samp, 1))
        self.y = np.sin(2*np.pi*self.x) + sigma *
np.random.normal(size=(num_samp, 1))

    def get_batch(self):
        choices = np.random.choice(self.index, size=BATCH_SIZE)

        return self.x[choices], self.y[choices].flatten()

x = tf.placeholder(tf.float32, [BATCH_SIZE, 1])
y = tf.placeholder(tf.float32, [BATCH_SIZE])

w = tf.get_variable('w', [NUM_FEATURES, 1], tf.float32,
                    tf.random_uniform_initializer(minval=0, maxval=1))
b = tf.get_variable('b', [], tf.float32,
tf.random_uniform_initializer(minval=0, maxval=1))
mu = tf.get_variable('mu', [NUM_FEATURES, 1], tf.float32,
                     tf.random_uniform_initializer(minval=0,
maxval=1))
sigma = tf.get_variable('sigma', [NUM_FEATURES, 1], tf.float32,
                        tf.random_uniform_initializer(minval=0,
maxval=1))

y_hat =  tf.reduce_sum(w*tf.exp(-(x-mu)**2/(sigma**2))) + b
loss = 0.5*tf.pow(y_hat - y, 2)

optim =
tf.train.GradientDescentOptimizer(learning_rate=0.01).minimize(loss)
init = tf.global_variables_initializer()
```



Fit 1
y
x
1.0
0.5
0.0
−0.5
−1.0
0.0
0.2
0.4
0.6
0.8
1.0



Bases for fit 1
y
x
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

```
sess = tf.Session()
sess.run(init)

data = Data()

for _ in tqdm(range(NUM_BATCHES)):
    for j in data.index:
        loss_np, _ = sess.run([loss, optim], feed_dict={x:
data.x[j].reshape(BATCH_SIZE,1), y: data.y[j]})

print("Parameter estimates:")
for var in tf.get_collection(tf.GraphKeys.TRAINABLE_VARIABLES):
    print(
        var.name.rstrip(":0"),
        np.array_str(np.array(sess.run(var)).flatten(), precision=3))

x_plt = np.linspace(0, 1, 1000)
y_plt = []
for i in x_plt:
    y_plt.append(sess.run(y_hat, feed_dict={x: i.reshape(BATCH_SIZE,
1)}))
y_plt = np.array(y_plt)

plt.subplot(1, 2, 1)
plt.plot(x_plt, np.sin(2*np.pi*x_plt), color='blue', linewidth=1)
plt.plot(x_plt, y_plt, color='red', linestyle='dashed', linewidth=1)
plt.scatter(data.x, data.y, color='green')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Fit 1')

plt.subplot(1, 2, 2)
for mu, sigma in zip(np.array(sess.run(mu)),
np.array(sess.run(sigma))):
    plt.plot(x_plt, np.exp(-(x_plt-mu)**2/(sigma**2)))
plt.xlabel('x')
plt.ylabel('y')
plt.title('Bases for fit 1')
plt.show()
```