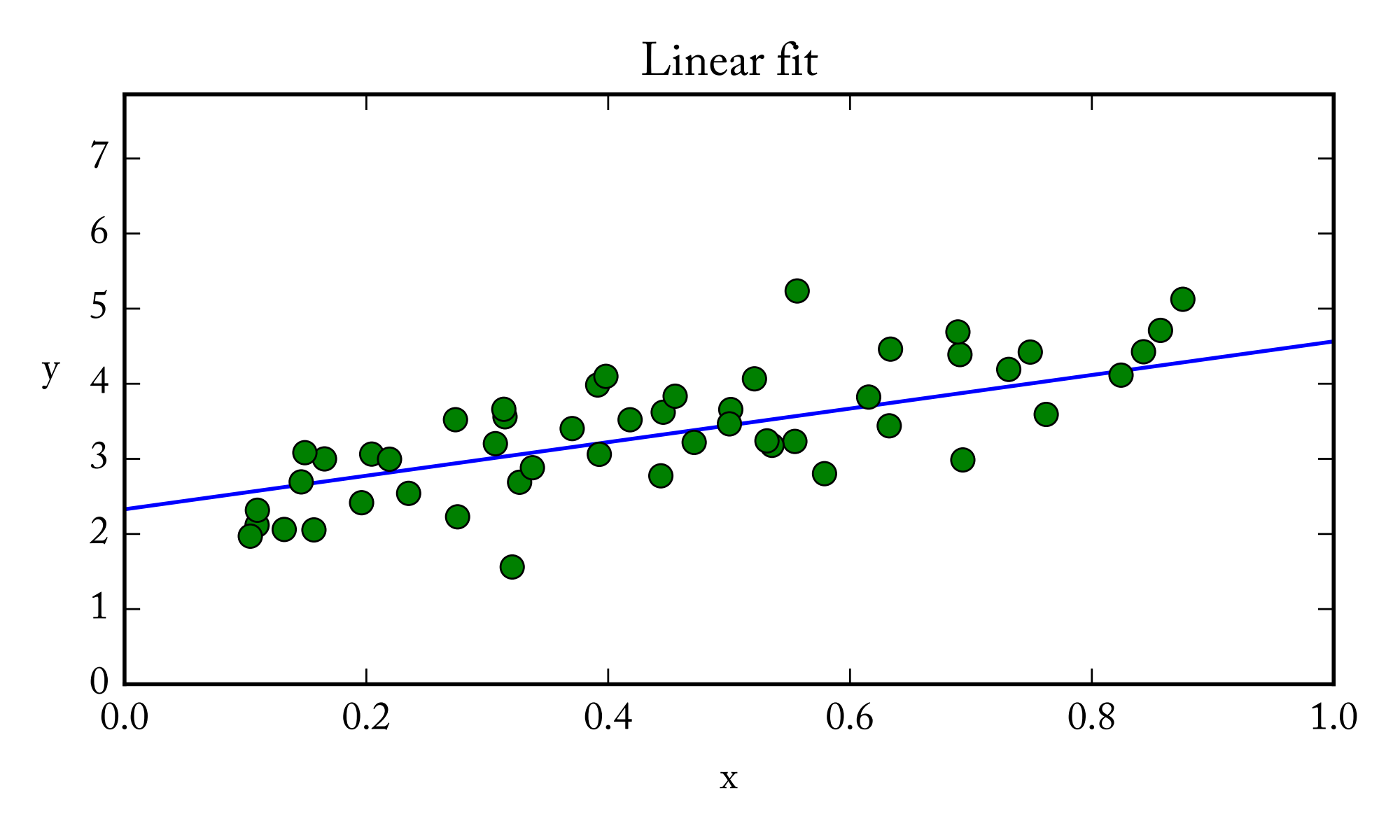
```python
#!/bin/env python3.8

"""
Example assignment. Author: Chris Curro
"""

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

from absl import app
from absl import flags
from tqdm import trange

font = {
    "family": "Adobe Caslon Pro",
    "size": 10,
}

matplotlib.style.use("classic")
matplotlib.rc("font", **font)

FLAGS = flags.FLAGS
flags.DEFINE_integer("num_features", 1, "Number of features in record")
flags.DEFINE_integer("num_samples", 50, "Number of samples in dataset")
flags.DEFINE_integer("batch_size", 16, "Number of samples in batch")
flags.DEFINE_integer("num_iters", 300, "Number of SGD iterations")
flags.DEFINE_float("learning_rate", 0.1, "Learning rate / step size for SGD")
flags.DEFINE_integer("random_seed", 31415, "Random seed")
flags.DEFINE_float("sigma_noise", 0.5, "Standard deviation of noise random variable")


class Data(object):
    def __init__(self, num_features, num_samp, sigma, random_seed):
        """
    Draw random weights and bias. Project vectors in R^NUM_FEATURES
    onto R with said weights and bias.
    """
        np.random.seed(random_seed)

        # We're going to learn these paramters
        self.w = np.random.randint(low=0, high=5, size=(num_features, 1))
        self.b = 2

        self.index = np.arange(num_samp)
        self.x = np.random.uniform(0.1, 0.9, size=(num_samp, num_features))
        self.y = self.x @ self.w + self.b + sigma * np.random.normal(size=(num_s
amp, 1))

    def get_batch(self, batch_size):
        """
    Select random subset of examples for training batch
    """
        choices = np.random.choice(self.index, size=batch_size)

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


class Model(tf.Module):
    def __init__(self, num_features):
        """
    A plain linear regression model with a bias term
    """
        self.w = tf.Variable(tf.random.normal(shape=[num_features, 1]))
        self.b = tf.Variable(tf.zeros(shape=[1, 1]))

    def __call__(self, x):
        return tf.squeeze(x @ self.w + self.b)
```



```python
def main(a):
    data = Data(
        FLAGS.num_features,
        FLAGS.num_samples,
        FLAGS.sigma_noise,
        FLAGS.random_seed,
    )
    model = Model(FLAGS.num_features)
    optimizer = tf.optimizers.SGD(learning_rate=FLAGS.learning_rate)

    bar = trange(FLAGS.num_iters)
    for i in bar:
        with tf.GradientTape() as tape:
            x, y = data.get_batch(FLAGS.batch_size)
            y_hat = model(x)
            loss = 0.5 * tf.reduce_mean((y_hat - y) ** 2)

        grads = tape.gradient(loss, model.trainable_variables)
        optimizer.apply_gradients(zip(grads, model.trainable_variables))

        bar.set_description(f"Loss @ {i} => {loss.numpy():0.6f}")
        bar.refresh()

    w = np.squeeze(data.w)
    w_hat = np.squeeze(model.w.numpy())

    # print out true values versus estimates
    print("w, w_hat")
    if FLAGS.num_features > 1:
        for a, b in zip(w, w_hat):
            print(f"{a}, {b:0.2f}")
    else:
        print(f"{w}, {w_hat:0.2f}")

    b_hat = np.squeeze(model.b.numpy())
    print("\nb, b_hat")
    print(f"{data.b}, {b_hat:0.2f}")

    if FLAGS.num_features > 1:
        # Only continue to plotting if x is a scalar
        exit(0)

    fig, ax = plt.subplots(1, 1, figsize=(5, 3), dpi=200)

    ax.set_title("Linear fit")
    ax.set_xlabel("x")
    ax.set_ylim(0, np.amax(data.y) * 1.5)
    h = ax.set_ylabel("y", labelpad=10)
    h.set_rotation(0)

    xs = np.linspace(0, 1, 10)
    xs = xs[:, np.newaxis]
    ax.plot(xs, np.squeeze(model(xs)), "-", np.squeeze(data.x), data.y, "o")

    plt.tight_layout()
    plt.savefig("fit.pdf")


if __name__ == "__main__":
    app.run(main)
```


Linear fit
y
x
0
1
2
3
4
5
6
7
0.0
0.2
0.4
0.6
0.8
1.0

```
fit: linear.py
        black linear.py
        flake8 --ignore=E,W linear.py
        python linear.py

pdf: fit
        a2ps linear.py -o linear.ps  --pro=color
        a2ps Makefile -o Makefile.ps  --pro=color
        ps2pdf linear.ps
        ps2pdf Makefile.ps
        gs -dBATCH -dNOPAUSE -q -sDEVICE=pdfwrite \
                -sOutputFile=example.pdf \
                linear.pdf fit.pdf Makefile.pdf
```