Machine Learning Homework 6

April 24, 2025

Gabe Morris

[1]:
```python
# toc
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt

from keras.api.models import Model
from keras.api.layers import Input, Dense
from keras.api.optimizers import Adam

plt.style.use('../maroon_ipynb.mplstyle')
tf.config.set_visible_devices([], 'GPU')
```

Contents

Problem 1

A function $u(x, y)$ is defined on a unit square $x \in [0, 1]$, $y \in [0, 1]$, and obeys the following partial differential equation:

$$\nabla^2 u(x, y) = e^{-x} \left(x - a + y^3 + by\right)$$

Where a and b are constants. The function $u(x, y)$ is also subject to the following Dirichlet boundary conditions:

$$\begin{gathered} u(0, y) = y^3 \\ u(1, y) = (1 + y^3)/e \\ u(x, 0) = xe^{-x} \\ u(x, 1) = e^{-x}(1 + x) \end{gathered}$$

For particular values of a and b the analytic solution is:

$$u(x, y) = e^{-x}(x + y^3)$$

Construct a physics-informed neural network (PINN) to determine the unknown constants a and b which give this solution. Try to minimize the number of points away from the boundary which explicitly use the analytic solution. Plot how the prediction of a and b evolves with the number of training epochs.

Solution

To solve this problem, we need to determine the unknown constants a and b in a partial differential equation (PDE) using a Physics-Informed Neural Network (PINN). The solution involves training a neural network to approximate the function $u(x, y)$ while satisfying the given boundary conditions and the PDE, which includes the constants a and b. These constants are treated as trainable variables in the model.

Generate Boundary and Collocation Points

Boundary points are sampled from the edges of the unit square, and their corresponding u values are computed using the given Dirichlet boundary conditions. Collocation points are randomly sampled from the interior of the unit square to enforce the PDE constraint.

[2]:
```
# Define some constant sizes
N_b = 100  # number of points per edge
N_f = 1000  # number of collocation points

# For x=0
x_b1 = np.zeros((N_b, 1))
y_b1 = np.linspace(0, 1, N_b).reshape(-1, 1)
u_b1 = y_b1**3
```

```python
# For x=1
x_b2 = np.ones((N_b, 1))
y_b2 = np.linspace(0, 1, N_b).reshape(-1, 1)
u_b2 = (1 + y_b2**3)/np.exp(1)

# For y=0
x_b3 = np.linspace(0, 1, N_b).reshape(-1, 1)
y_b3 = np.zeros((N_b, 1))
u_b3 = x_b3*np.exp(-x_b3)

# For y=1
x_b4 = np.linspace(0, 1, N_b).reshape(-1, 1)
y_b4 = np.ones((N_b, 1))
u_b4 = np.exp(-x_b4)*(1 + x_b4)

# Combine all boundary points
X_bc = np.vstack([
    np.hstack([x_b1, y_b1]),
    np.hstack([x_b2, y_b2]),
    np.hstack([x_b3, y_b3]),
    np.hstack([x_b4, y_b4])
])
u_bc = np.vstack([u_b1, u_b2, u_b3, u_b4])

# Generate collocation points
x_f = np.random.rand(N_f, 1)
y_f = np.random.rand(N_f, 1)
X_pde = np.hstack([x_f, y_f])

# Convert to TensorFlow tensors
X_bc_tf = tf.convert_to_tensor(X_bc, dtype=tf.float32)
u_bc_tf = tf.convert_to_tensor(u_bc, dtype=tf.float32)
X_pde_tf = tf.convert_to_tensor(X_pde, dtype=tf.float32)
```

Build the PINN Model

[3]:
```python
def create_model():
    inputs = Input(shape=(2,))
    x = Dense(64, activation='tanh')(inputs)
    x = Dense(64, activation='tanh')(x)
    x = Dense(64, activation='tanh')(x)
    outputs = Dense(1)(x)
    model_ = Model(inputs=inputs, outputs=outputs)
    return model_


model = create_model()
```

```
# Initialize trainable variables a and b
a = tf.Variable(1.5, dtype=tf.float32, name='a')
b = tf.Variable(6.5, dtype=tf.float32, name='b')

# Define the optimizer
optimizer = Adam(learning_rate=0.001)
```

Training Step

The total loss (sum of boundary, PDE losses, and analytical/data loss) is minimized using the Adam optimizer. The gradients of the total loss with respect to the network's parameters and the constants a and b are computed and used to update these variables.

[4]:
```
pde_loss_weight = 1  # weight to emphasize PDE constraint
analytic_loss_weight = 1  # weight to emphasize the analytic solution

@tf.function
def train_step(X_bc_, u_bc_, X_pde_):
    with tf.GradientTape() as tape:
        # BC loss
        u_pred_bc = model(X_bc_, training=True)
        bc_loss_ = tf.reduce_mean((u_bc_ - u_pred_bc)**2)

        # PDE loss
        with tf.GradientTape(persistent=True) as tape2:
            tape2.watch(X_pde_)
            u_pred_pde = model(X_pde_, training=True)
            grad_u = tape2.gradient(u_pred_pde, X_pde_)
            u_x = grad_u[:, 0:1]
            u_y = grad_u[:, 1:2]
        u_xx = tape2.gradient(u_x, X_pde_)[:, 0:1]
        u_yy = tape2.gradient(u_y, X_pde_)[:, 1:2]
        del tape2

        x_pde = X_pde_[:, 0:1]
        y_pde = X_pde_[:, 1:2]
        rhs = tf.exp(-x_pde)*(x_pde - a + y_pde**3 + b*y_pde)
        PDE_resid = u_xx + u_yy - rhs
        pde_loss_ = tf.reduce_mean(PDE_resid**2)

        # Analytic loss
        u_analytic = tf.exp(-x_pde)*(x_pde + y_pde**3)
        analytic_loss_ = tf.reduce_mean((u_analytic - u_pred_pde)**2)

        total_loss_ = bc_loss_ + pde_loss_weight*pde_loss_ +␣
↪analytic_loss_weight*analytic_loss_
```

```python
    grads = tape.gradient(total_loss_, model.trainable_variables + [a, b])
    optimizer.apply_gradients(zip(grads, model.trainable_variables + [a, b]))
    return bc_loss_, pde_loss_, analytic_loss_, total_loss_
```

Now we can finish training and tracking the values of a and b.

[5]:

```python
epochs = 50_000
a_history, b_history = [], []
loss_history = []

for epoch in range(1, epochs + 1):
    # Resample interior
    x_f = np.random.rand(N_f, 1)
    y_f = np.random.rand(N_f, 1)
    X_pde_tf = tf.convert_to_tensor(np.hstack([x_f, y_f]), dtype=tf.float32)

    bc_loss, pde_loss, analytical_loss, total_loss = train_step(X_bc_tf,␣
 ↪u_bc_tf, X_pde_tf)
    a_history.append(a.numpy())
    b_history.append(b.numpy())

    if epoch%500 == 0:
        print(f"Epoch {epoch:5d}  |  BC_loss = {bc_loss:.3e}  PDE_loss =␣
 ↪{pde_loss:.3e} A_loss = {analytical_loss:.3e} a = {a.numpy():.4f}  b = {b.
 ↪numpy():.4f}")

epoch_array = np.arange(1, epochs + 1)
plt.plot(epoch_array, a_history, label='a')
plt.plot(epoch_array, b_history, label='b')
plt.axhline(y=2, ls='--', color='darkgrey')
plt.axhline(y=6, ls='--', color='darkgrey')
plt.xlabel('Epochs')
plt.ylabel('Value')
plt.legend()
plt.show()
```

```
WARNING:tensorflow:Calling GradientTape.gradient on a persistent tape inside its
context is significantly less efficient than calling it outside the context (it
causes the gradient ops to be recorded on the tape, leading to increased CPU and
memory usage). Only call GradientTape.gradient inside the context if you
actually want to trace the gradient in order to compute higher order
derivatives.
Epoch   500  |  BC_loss = 2.523e-03  PDE_loss = 2.926e-03 A_loss = 8.478e-04 a =
1.5470  b = 6.4472
Epoch  1000  |  BC_loss = 1.960e-04  PDE_loss = 4.811e-04 A_loss = 4.051e-04 a =
1.5489  b = 6.4461
Epoch  1500  |  BC_loss = 2.281e-04  PDE_loss = 1.315e-04 A_loss = 1.560e-04 a =
```

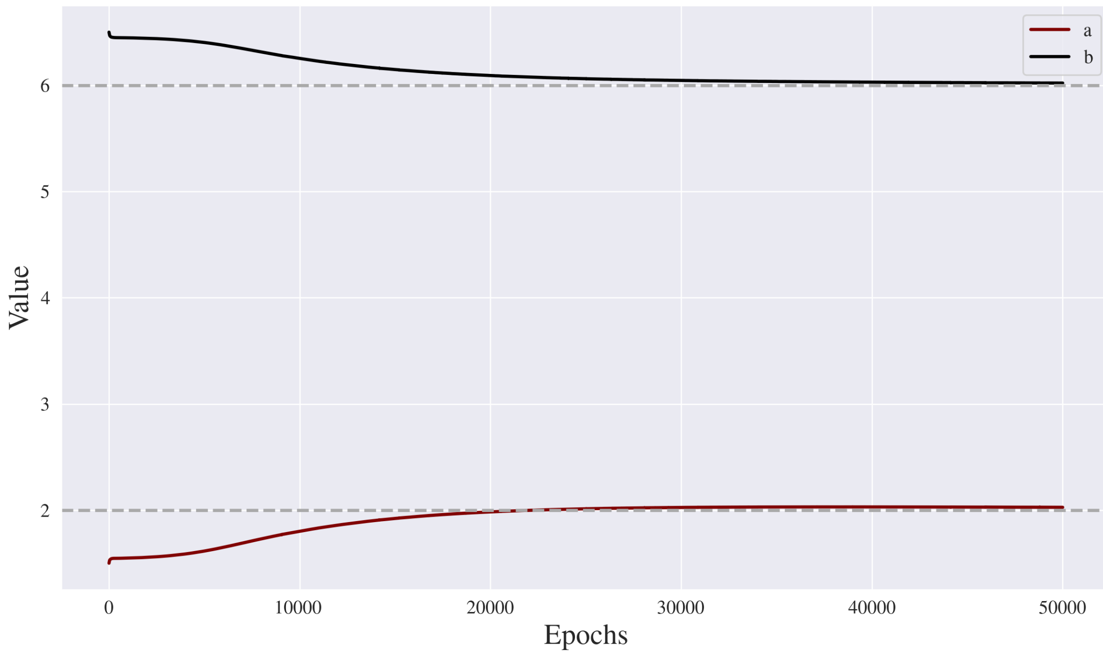
```
1.5520  b = 6.4442
Epoch  2000  |  BC_loss = 1.504e-04  PDE_loss = 7.817e-05 A_loss = 1.994e-04 a =
1.5562  b = 6.4416
Epoch  2500  |  BC_loss = 8.924e-05  PDE_loss = 1.404e-04 A_loss = 3.203e-04 a =
1.5616  b = 6.4382
Epoch  3000  |  BC_loss = 1.639e-04  PDE_loss = 2.183e-04 A_loss = 1.925e-04 a =
1.5686  b = 6.4337
Epoch  3500  |  BC_loss = 9.720e-05  PDE_loss = 5.953e-05 A_loss = 2.488e-04 a =
1.5772  b = 6.4280
Epoch  4000  |  BC_loss = 1.297e-04  PDE_loss = 1.516e-04 A_loss = 1.543e-04 a =
1.5878  b = 6.4210
Epoch  4500  |  BC_loss = 5.475e-04  PDE_loss = 1.046e-03 A_loss = 1.558e-04 a =
1.6006  b = 6.4123
Epoch  5000  |  BC_loss = 6.170e-05  PDE_loss = 1.109e-04 A_loss = 2.890e-04 a =
1.6154  b = 6.4021
Epoch  5500  |  BC_loss = 1.190e-04  PDE_loss = 1.494e-04 A_loss = 1.240e-04 a =
1.6321  b = 6.3900
Epoch  6000  |  BC_loss = 8.887e-05  PDE_loss = 1.317e-05 A_loss = 1.369e-04 a =
1.6506  b = 6.3762
Epoch  6500  |  BC_loss = 1.257e-04  PDE_loss = 2.180e-05 A_loss = 8.735e-05 a =
1.6702  b = 6.3614
Epoch  7000  |  BC_loss = 5.814e-05  PDE_loss = 1.132e-04 A_loss = 1.206e-04 a =
1.6904  b = 6.3454
Epoch  7500  |  BC_loss = 3.246e-04  PDE_loss = 1.850e-05 A_loss = 8.467e-05 a =
1.7107  b = 6.3290
Epoch  8000  |  BC_loss = 5.962e-05  PDE_loss = 1.141e-05 A_loss = 8.213e-05 a =
1.7306  b = 6.3125
Epoch  8500  |  BC_loss = 5.639e-05  PDE_loss = 7.340e-05 A_loss = 6.211e-05 a =
1.7500  b = 6.2963
Epoch  9000  |  BC_loss = 2.478e-05  PDE_loss = 8.141e-06 A_loss = 1.171e-04 a =
1.7684  b = 6.2806
Epoch  9500  |  BC_loss = 1.366e-04  PDE_loss = 3.675e-05 A_loss = 3.762e-05 a =
1.7849  b = 6.2665
Epoch 10000  |  BC_loss = 9.359e-05  PDE_loss = 2.256e-05 A_loss = 2.591e-05 a =
1.8016  b = 6.2521
Epoch 10500  |  BC_loss = 4.561e-05  PDE_loss = 4.385e-05 A_loss = 2.932e-05 a =
1.8174  b = 6.2386
Epoch 11000  |  BC_loss = 2.028e-05  PDE_loss = 2.000e-05 A_loss = 6.466e-05 a =
1.8326  b = 6.2252
Epoch 11500  |  BC_loss = 8.077e-05  PDE_loss = 1.023e-05 A_loss = 2.088e-05 a =
1.8461  b = 6.2138
Epoch 12000  |  BC_loss = 1.603e-05  PDE_loss = 5.899e-05 A_loss = 4.645e-05 a =
1.8592  b = 6.2026
Epoch 12500  |  BC_loss = 4.788e-05  PDE_loss = 9.831e-06 A_loss = 1.413e-05 a =
1.8713  b = 6.1919
Epoch 13000  |  BC_loss = 2.727e-05  PDE_loss = 4.291e-04 A_loss = 9.957e-05 a =
1.8830  b = 6.1818
Epoch 13500  |  BC_loss = 1.363e-05  PDE_loss = 7.390e-06 A_loss = 2.126e-05 a =
```

```
1.8941  b = 6.1718
Epoch 14000  |  BC_loss = 1.815e-05  PDE_loss = 4.107e-05 A_loss = 1.096e-05 a =
1.9040  b = 6.1633
Epoch 14500  |  BC_loss = 1.698e-05  PDE_loss = 5.401e-06 A_loss = 8.622e-06 a =
1.9132  b = 6.1552
Epoch 15000  |  BC_loss = 1.292e-05  PDE_loss = 4.562e-06 A_loss = 5.011e-05 a =
1.9221  b = 6.1474
Epoch 15500  |  BC_loss = 7.260e-06  PDE_loss = 3.393e-06 A_loss = 1.083e-05 a =
1.9301  b = 6.1401
Epoch 16000  |  BC_loss = 1.520e-04  PDE_loss = 2.280e-04 A_loss = 8.999e-05 a =
1.9380  b = 6.1333
Epoch 16500  |  BC_loss = 2.419e-04  PDE_loss = 8.008e-04 A_loss = 3.620e-04 a =
1.9452  b = 6.1268
Epoch 17000  |  BC_loss = 6.696e-05  PDE_loss = 9.631e-05 A_loss = 3.246e-05 a =
1.9521  b = 6.1205
Epoch 17500  |  BC_loss = 1.578e-05  PDE_loss = 1.510e-05 A_loss = 3.904e-06 a =
1.9586  b = 6.1144
Epoch 18000  |  BC_loss = 5.701e-06  PDE_loss = 3.523e-06 A_loss = 3.360e-06 a =
1.9644  b = 6.1091
Epoch 18500  |  BC_loss = 2.878e-05  PDE_loss = 8.014e-05 A_loss = 8.526e-06 a =
1.9698  b = 6.1040
Epoch 19000  |  BC_loss = 6.035e-06  PDE_loss = 4.117e-06 A_loss = 2.163e-06 a =
1.9747  b = 6.0997
Epoch 19500  |  BC_loss = 2.787e-06  PDE_loss = 6.593e-05 A_loss = 6.186e-06 a =
1.9794  b = 6.0952
Epoch 20000  |  BC_loss = 1.272e-04  PDE_loss = 3.409e-05 A_loss = 1.700e-04 a =
1.9838  b = 6.0911
Epoch 20500  |  BC_loss = 2.037e-05  PDE_loss = 2.100e-04 A_loss = 3.687e-05 a =
1.9877  b = 6.0873
Epoch 21000  |  BC_loss = 2.078e-06  PDE_loss = 8.640e-06 A_loss = 2.169e-06 a =
1.9915  b = 6.0835
Epoch 21500  |  BC_loss = 1.174e-04  PDE_loss = 5.698e-04 A_loss = 1.518e-04 a =
1.9948  b = 6.0803
Epoch 22000  |  BC_loss = 4.739e-05  PDE_loss = 5.263e-05 A_loss = 3.464e-05 a =
1.9982  b = 6.0768
Epoch 22500  |  BC_loss = 2.504e-06  PDE_loss = 5.628e-06 A_loss = 4.995e-06 a =
2.0012  b = 6.0738
Epoch 23000  |  BC_loss = 3.961e-04  PDE_loss = 3.128e-04 A_loss = 3.713e-04 a =
2.0041  b = 6.0708
Epoch 23500  |  BC_loss = 1.909e-06  PDE_loss = 1.743e-06 A_loss = 5.357e-07 a =
2.0066  b = 6.0682
Epoch 24000  |  BC_loss = 3.613e-06  PDE_loss = 6.637e-05 A_loss = 1.112e-06 a =
2.0088  b = 6.0658
Epoch 24500  |  BC_loss = 2.808e-06  PDE_loss = 6.802e-06 A_loss = 4.889e-06 a =
2.0109  b = 6.0635
Epoch 25000  |  BC_loss = 5.178e-06  PDE_loss = 1.301e-05 A_loss = 2.124e-06 a =
2.0129  b = 6.0613
Epoch 25500  |  BC_loss = 7.016e-06  PDE_loss = 5.978e-05 A_loss = 6.861e-06 a =
```

```
2.0150  b = 6.0587
Epoch 26000  |  BC_loss = 7.190e-05  PDE_loss = 2.353e-05 A_loss = 8.153e-05 a =
2.0166  b = 6.0569
Epoch 26500  |  BC_loss = 6.970e-07  PDE_loss = 1.213e-06 A_loss = 2.852e-07 a =
2.0181  b = 6.0550
Epoch 27000  |  BC_loss = 3.775e-05  PDE_loss = 8.398e-05 A_loss = 3.638e-05 a =
2.0195  b = 6.0533
Epoch 27500  |  BC_loss = 4.501e-06  PDE_loss = 2.283e-05 A_loss = 3.739e-06 a =
2.0210  b = 6.0514
Epoch 28000  |  BC_loss = 1.483e-05  PDE_loss = 5.684e-06 A_loss = 1.680e-05 a =
2.0219  b = 6.0503
Epoch 28500  |  BC_loss = 1.950e-06  PDE_loss = 1.262e-06 A_loss = 2.143e-06 a =
2.0230  b = 6.0487
Epoch 29000  |  BC_loss = 1.154e-06  PDE_loss = 3.956e-06 A_loss = 6.364e-07 a =
2.0239  b = 6.0475
Epoch 29500  |  BC_loss = 4.962e-07  PDE_loss = 1.571e-06 A_loss = 2.547e-07 a =
2.0248  b = 6.0460
Epoch 30000  |  BC_loss = 2.103e-05  PDE_loss = 1.549e-04 A_loss = 1.681e-05 a =
2.0256  b = 6.0448
Epoch 30500  |  BC_loss = 3.827e-06  PDE_loss = 1.435e-05 A_loss = 2.577e-06 a =
2.0266  b = 6.0431
Epoch 31000  |  BC_loss = 4.923e-05  PDE_loss = 8.519e-06 A_loss = 4.839e-05 a =
2.0272  b = 6.0422
Epoch 31500  |  BC_loss = 1.960e-04  PDE_loss = 6.677e-04 A_loss = 1.599e-04 a =
2.0277  b = 6.0412
Epoch 32000  |  BC_loss = 1.201e-06  PDE_loss = 5.419e-06 A_loss = 6.793e-07 a =
2.0282  b = 6.0402
Epoch 32500  |  BC_loss = 5.200e-07  PDE_loss = 1.414e-06 A_loss = 2.995e-07 a =
2.0286  b = 6.0394
Epoch 33000  |  BC_loss = 4.013e-06  PDE_loss = 1.055e-04 A_loss = 1.650e-06 a =
2.0290  b = 6.0385
Epoch 33500  |  BC_loss = 3.822e-06  PDE_loss = 1.401e-05 A_loss = 4.400e-06 a =
2.0293  b = 6.0377
Epoch 34000  |  BC_loss = 9.144e-05  PDE_loss = 3.267e-03 A_loss = 8.340e-05 a =
2.0298  b = 6.0366
Epoch 34500  |  BC_loss = 1.226e-06  PDE_loss = 5.008e-05 A_loss = 3.622e-07 a =
2.0299  b = 6.0360
Epoch 35000  |  BC_loss = 3.813e-06  PDE_loss = 2.455e-06 A_loss = 3.015e-06 a =
2.0301  b = 6.0352
Epoch 35500  |  BC_loss = 1.848e-05  PDE_loss = 3.194e-04 A_loss = 1.149e-05 a =
2.0302  b = 6.0346
Epoch 36000  |  BC_loss = 1.354e-05  PDE_loss = 4.739e-06 A_loss = 1.267e-05 a =
2.0304  b = 6.0338
Epoch 36500  |  BC_loss = 1.657e-06  PDE_loss = 4.506e-05 A_loss = 5.911e-07 a =
2.0306  b = 6.0329
Epoch 37000  |  BC_loss = 2.053e-04  PDE_loss = 1.175e-05 A_loss = 1.949e-04 a =
2.0306  b = 6.0324
Epoch 37500  |  BC_loss = 5.944e-05  PDE_loss = 2.986e-05 A_loss = 5.857e-05 a =
```

```
2.0307  b = 6.0317
Epoch 38000  |  BC_loss = 1.815e-06  PDE_loss = 4.910e-05 A_loss = 9.471e-07 a =
2.0306  b = 6.0312
Epoch 38500  |  BC_loss = 1.659e-06  PDE_loss = 3.781e-06 A_loss = 1.659e-06 a =
2.0306  b = 6.0306
Epoch 39000  |  BC_loss = 5.702e-06  PDE_loss = 1.697e-04 A_loss = 2.756e-06 a =
2.0308  b = 6.0297
Epoch 39500  |  BC_loss = 2.551e-04  PDE_loss = 1.003e-04 A_loss = 2.700e-04 a =
2.0307  b = 6.0292
Epoch 40000  |  BC_loss = 3.111e-07  PDE_loss = 3.129e-06 A_loss = 1.517e-07 a =
2.0305  b = 6.0290
Epoch 40500  |  BC_loss = 1.205e-05  PDE_loss = 7.638e-06 A_loss = 1.077e-05 a =
2.0305  b = 6.0283
Epoch 41000  |  BC_loss = 4.550e-05  PDE_loss = 3.737e-04 A_loss = 3.486e-05 a =
2.0303  b = 6.0279
Epoch 41500  |  BC_loss = 2.956e-06  PDE_loss = 5.073e-06 A_loss = 2.116e-06 a =
2.0302  b = 6.0274
Epoch 42000  |  BC_loss = 3.386e-07  PDE_loss = 2.169e-06 A_loss = 1.152e-07 a =
2.0301  b = 6.0267
Epoch 42500  |  BC_loss = 1.151e-06  PDE_loss = 1.495e-06 A_loss = 5.762e-07 a =
2.0301  b = 6.0263
Epoch 43000  |  BC_loss = 7.591e-06  PDE_loss = 2.480e-05 A_loss = 4.241e-06 a =
2.0300  b = 6.0256
Epoch 43500  |  BC_loss = 5.173e-07  PDE_loss = 3.389e-06 A_loss = 7.160e-07 a =
2.0297  b = 6.0254
Epoch 44000  |  BC_loss = 5.312e-07  PDE_loss = 2.657e-06 A_loss = 1.906e-07 a =
2.0295  b = 6.0250
Epoch 44500  |  BC_loss = 1.101e-06  PDE_loss = 1.087e-06 A_loss = 5.777e-07 a =
2.0294  b = 6.0245
Epoch 45000  |  BC_loss = 1.375e-05  PDE_loss = 2.747e-06 A_loss = 1.190e-05 a =
2.0293  b = 6.0239
Epoch 45500  |  BC_loss = 3.930e-07  PDE_loss = 8.925e-07 A_loss = 5.347e-07 a =
2.0290  b = 6.0237
Epoch 46000  |  BC_loss = 6.970e-06  PDE_loss = 4.998e-05 A_loss = 2.503e-06 a =
2.0294  b = 6.0218
Epoch 46500  |  BC_loss = 2.383e-07  PDE_loss = 1.339e-06 A_loss = 7.805e-08 a =
2.0288  b = 6.0223
Epoch 47000  |  BC_loss = 7.573e-07  PDE_loss = 1.431e-05 A_loss = 5.579e-07 a =
2.0285  b = 6.0223
Epoch 47500  |  BC_loss = 1.762e-07  PDE_loss = 1.313e-06 A_loss = 1.387e-07 a =
2.0282  b = 6.0220
Epoch 48000  |  BC_loss = 3.187e-05  PDE_loss = 5.253e-04 A_loss = 1.683e-05 a =
2.0281  b = 6.0214
Epoch 48500  |  BC_loss = 1.834e-05  PDE_loss = 4.986e-06 A_loss = 1.618e-05 a =
2.0279  b = 6.0211
Epoch 49000  |  BC_loss = 1.983e-07  PDE_loss = 1.061e-06 A_loss = 2.298e-07 a =
2.0277  b = 6.0205
Epoch 49500  |  BC_loss = 2.709e-05  PDE_loss = 6.004e-06 A_loss = 2.443e-05 a =
```

```
2.0273  b = 6.0205
Epoch 50000  |  BC_loss = 3.119e-06  PDE_loss = 3.820e-05 A_loss = 1.534e-06 a =
2.0270  b = 6.0202
```



The above shows that the values for a and b converge to the expected values of 2 and 6. The analytical loss in the real world could be substituted for data loss.