Physics-Informed Neural Networks (PINNs) Implementation

This notebook implements PINNs for differential equations, including:

1. Lorenz-1960 System
2. Harmonic Oscillator
3. Hard Constraints

In [ ]:

```
!apt-get install texlive-generic-recommended
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package texlive-generic-recommended
```

In [ ]:

```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc  texlive-generic-recommended

!pip install pypandoc    texlive-generic-recommended
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package texlive-generic-recommended
Requirement already satisfied: pypandoc in /usr/local/lib/python3.11/dist-packages (1.15)
ERROR: Could not find a version that satisfies the requirement texlive-generic-recommended (from versions: none)
ERROR: No matching distribution found for texlive-generic-recommended
```

In [ ]:

```
# Install required packages
!pip install pyDOE
!pip install tensorflow
!pip install matplotlib
!pip install numpy
!pip install scipy
```

```
Collecting pyDOE
  Downloading pyDOE-0.3.8.zip (22 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from pyDOE) (1.26.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from pyDOE) (1.13.1)
Building wheels for collected packages: pyDOE
  Building wheel for pyDOE (setup.py) ... done
  Created wheel for pyDOE: filename=pyDOE-0.3.8-py3-none-any.whl size=18170 sha256=5e4e980b5715943ff36e7ebb8e431a2e
cbc0bf3e99fb971b2efa1c88afcea96b
  Stored in directory: /root/.cache/pip/wheels/84/20/8c/8bd43ba42b0b6d39ace1219d6da1576e0dac81b12265c4762e
Successfully built pyDOE
Installing collected packages: pyDOE
Successfully installed pyDOE-0.3.8
Requirement already satisfied: tensorflow in /usr/local/lib/python3.11/dist-packages (2.18.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2
5.1.24)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.11/dist-packages (from
tensorflow) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(0.2.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1
8.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(3.4.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from tensorflow) (24.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 in
/usr/local/lib/python3.11/dist-packages (from tensorflow) (4.25.6)
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(2.32.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from tensorflow) (75.1.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(2.5.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.11/dist-packages (from tensorflo
w) (4.12.2)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.2)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(1.70.0)
Requirement already satisfied: tensorboard<2.19,>=2.18 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(2.18.0)
Requirement already satisfied: keras>=3.5.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.8.0)
Requirement already satisfied: numpy<2.1.0,>=1.26.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(1.26.4)
Requirement already satisfied: h5py>=3.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.12.1)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow)
(0.4.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.11/dist-packages (fro
m tensorflow) (0.37.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.11/dist-packages (from astunparse>=
1.6.0->tensorflow) (0.45.1)
Requirement already satisfied: rich in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow) (1
3.9.4)
Requirement already satisfied: namex in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow)
(0.0.8)
Requirement already satisfied: optree in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow)
(0.14.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests<
3,>=2.21.0->tensorflow) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->t
ensorflow) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.2
1.0->tensorflow) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.2
1.0->tensorflow) (2025.1.31)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,>=
2.18->tensorflow) (3.7)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.11/dist-packages (fr
om tensorboard<2.19,>=2.18->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,>=
2.18->tensorflow) (3.1.3)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.11/dist-packages (from werkzeug>=1.0.1->
tensorboard<2.19,>=2.18->tensorflow) (3.0.2)
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich->keras>=
3.5.0->tensorflow) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich->keras
>=3.5.0->tensorflow) (2.18.0)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->r
ich->keras>=3.5.0->tensorflow) (0.1.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib)
(1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.5
5.8)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib)
(1.4.8)
Requirement already satisfied: numpy>=1.23 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.26.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib)
(3.2.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib)
(2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matp
lotlib) (1.17.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (1.26.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (1.13.1)
Requirement already satisfied: numpy<2.3,>=1.22.4 in /usr/local/lib/python3.11/dist-packages (from scipy) (1.26.4)
```

In [ ]:

```python
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp
from pyDOE import lhs

# Enable GPU acceleration if available
print('TensorFlow version:', tf.__version__)
print('GPU Available:', tf.test.is_gpu_available())
```

```
WARNING:tensorflow:From <ipython-input-2-6786947b212f>:9: is_gpu_available (from tensorflow.python.framework.test_u
til) is deprecated and will be removed in a future version.
Instructions for updating:
Use `tf.config.list_physical_devices('GPU')` instead.
TensorFlow version: 2.18.0
GPU Available: False
```

1. Base PINN Architecture

In [ ]:

```python
def build_model(nr_units=20, nr_layers=4, output_dim=1):
    inp = tf.keras.layers.Input(shape=(1,))
    x = inp

    for _ in range(nr_layers):
        x = tf.keras.layers.Dense(nr_units, activation='tanh')(x)

    out = tf.keras.layers.Dense(output_dim, activation='linear')(x)
    return tf.keras.models.Model(inp, out)

def defineCollocationPoints(t_bdry, N_de=100):
    return t_bdry[0] + (t_bdry[1] - t_bdry[0])*lhs(1, N_de)
```

2. Lorenz-1960 Implementation

In [ ]:

```python
# Constants for the Lorenz-1960 model
k, l = 1, 2
x0, y0, z0 = 1, 0.5, 1  # Initial conditions

@tf.function
def compute_lorenz_loss(t, model, gamma=1):
    with tf.GradientTape() as tape:
        with tf.GradientTape(persistent=True) as tape2:
            tape2.watch(t)
            pred = model(t)
            x_pred, y_pred, z_pred = tf.split(pred, 3, axis=1)

        dx_dt = tape2.gradient(x_pred, t)
        dy_dt = tape2.gradient(y_pred, t)
        dz_dt = tape2.gradient(z_pred, t)

        eq1 = dx_dt - k * l * ((1/k**2 + l**2) - (1/k**2)) * y_pred * z_pred
        eq2 = dy_dt - k * l * ((1/l**2) - (1/k**2 + l**2)) * x_pred * z_pred
        eq3 = dz_dt - (k * l**2) * ((1/k**2) - (1/l**2)) * x_pred * y_pred

        DEloss = tf.reduce_mean(eq1**2 + eq2**2 + eq3**2)
        u0_pred = model(tf.constant([[0.0]], dtype=tf.float32))
        IVloss = tf.reduce_mean((u0_pred[0,0] - x0)**2 + (u0_pred[0,1] - y0)**2 + (u0_pred[0,2] - z0)**2)

        loss = DEloss + gamma * IVloss

    grads = tape.gradient(loss, model.trainable_variables)
    return loss, grads, DEloss, IVloss
```

3. Harmonic Oscillator Implementation

In [ ]:

```
# Constants for Harmonic Oscillator
m = 1.0
k_spring = 2.0
omega = np.sqrt(k_spring/m)
T = 2*np.pi/omega

# Initial conditions
u0 = 1.0
v0 = 1.0

@tf.function
def train_first_order(t, model, gamma=1):
    with tf.GradientTape() as tape:
        with tf.GradientTape(persistent=True) as tape2:
            tape2.watch(t)
            UV = model(t)
            u = UV[:, 0:1]
            v = UV[:, 1:2]

        du_dt = tape2.gradient(u, t)
        dv_dt = tape2.gradient(v, t)

        loss_du = tf.reduce_mean((du_dt - v)**2)
        loss_dv = tf.reduce_mean((dv_dt + (k_spring/m)*u)**2)

        UV0 = model(tf.constant([[0.0]]))
        loss_u0 = tf.reduce_mean((UV0[:, 0] - u0)**2)
        loss_v0 = tf.reduce_mean((UV0[:, 1] - v0)**2)

        total_loss = loss_du + loss_dv + gamma*(loss_u0 + loss_v0)

    grads = tape.gradient(total_loss, model.trainable_variables)
    return total_loss, grads
```

4. Training Utilities

In [ ]:

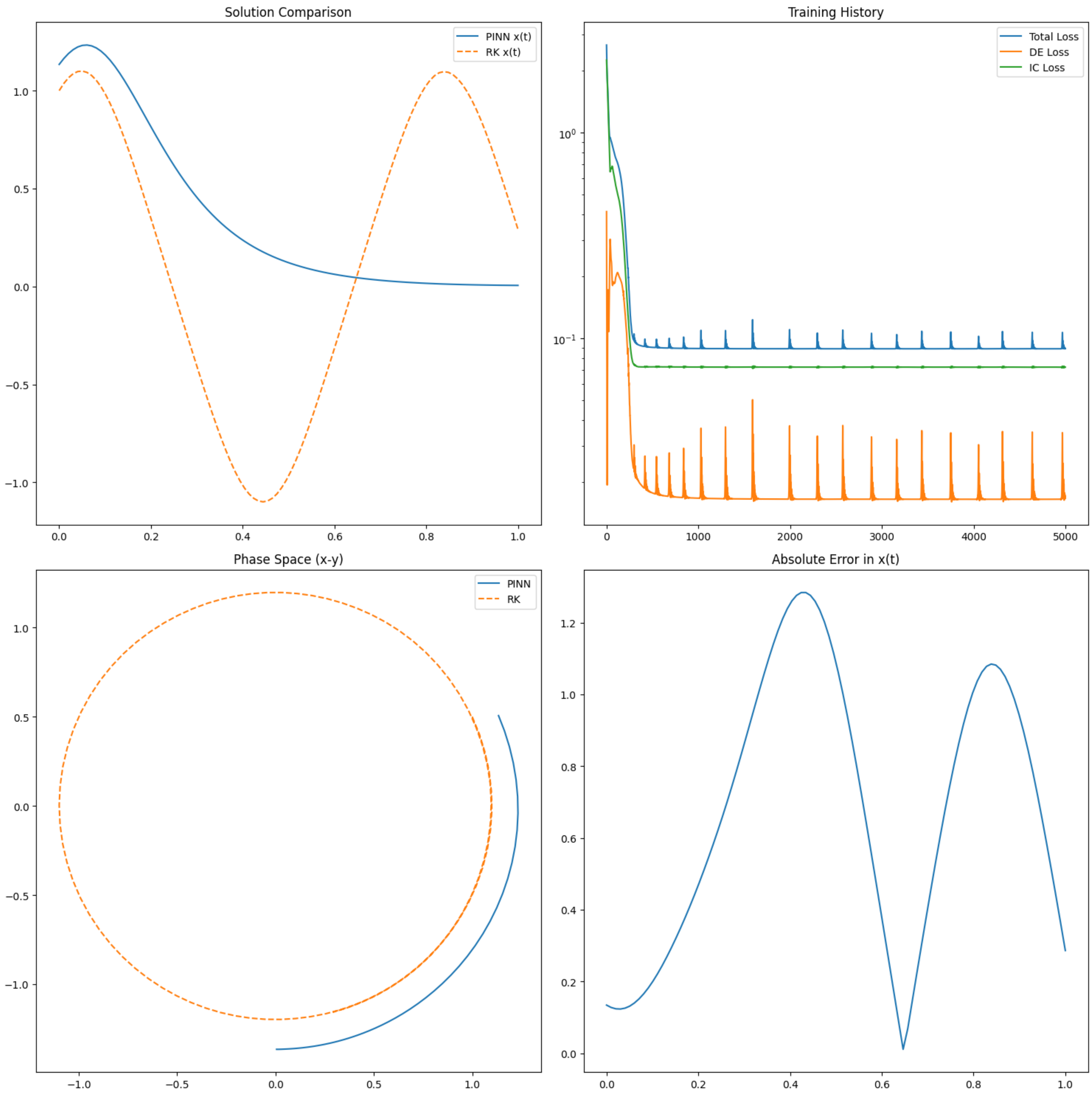
```
def train_model(de_points, model, loss_fn, epochs=5000, batch_size=100, learning_rate=1e-3):
    ds = tf.data.Dataset.from_tensor_slices(de_points.astype(np.float32))
    ds = ds.shuffle(1000).batch(batch_size)

    optimizer = tf.keras.optimizers.Adam(learning_rate=learning_rate)
    history = {'loss': [], 'de_loss': [], 'iv_loss': []}

    for epoch in range(epochs):
        epoch_loss = 0
        for batch in ds:
            loss, grads, de_loss, iv_loss = loss_fn(batch, model)
            optimizer.apply_gradients(zip(grads, model.trainable_variables))

            history['loss'].append(float(loss))
            history['de_loss'].append(float(de_loss))
            history['iv_loss'].append(float(iv_loss))

        if epoch % 500 == 0:
            print(f'Epoch {epoch}: Loss = {history["loss"][-1]:.4e}')

    return history
```

5. Experiment: Lorenz System

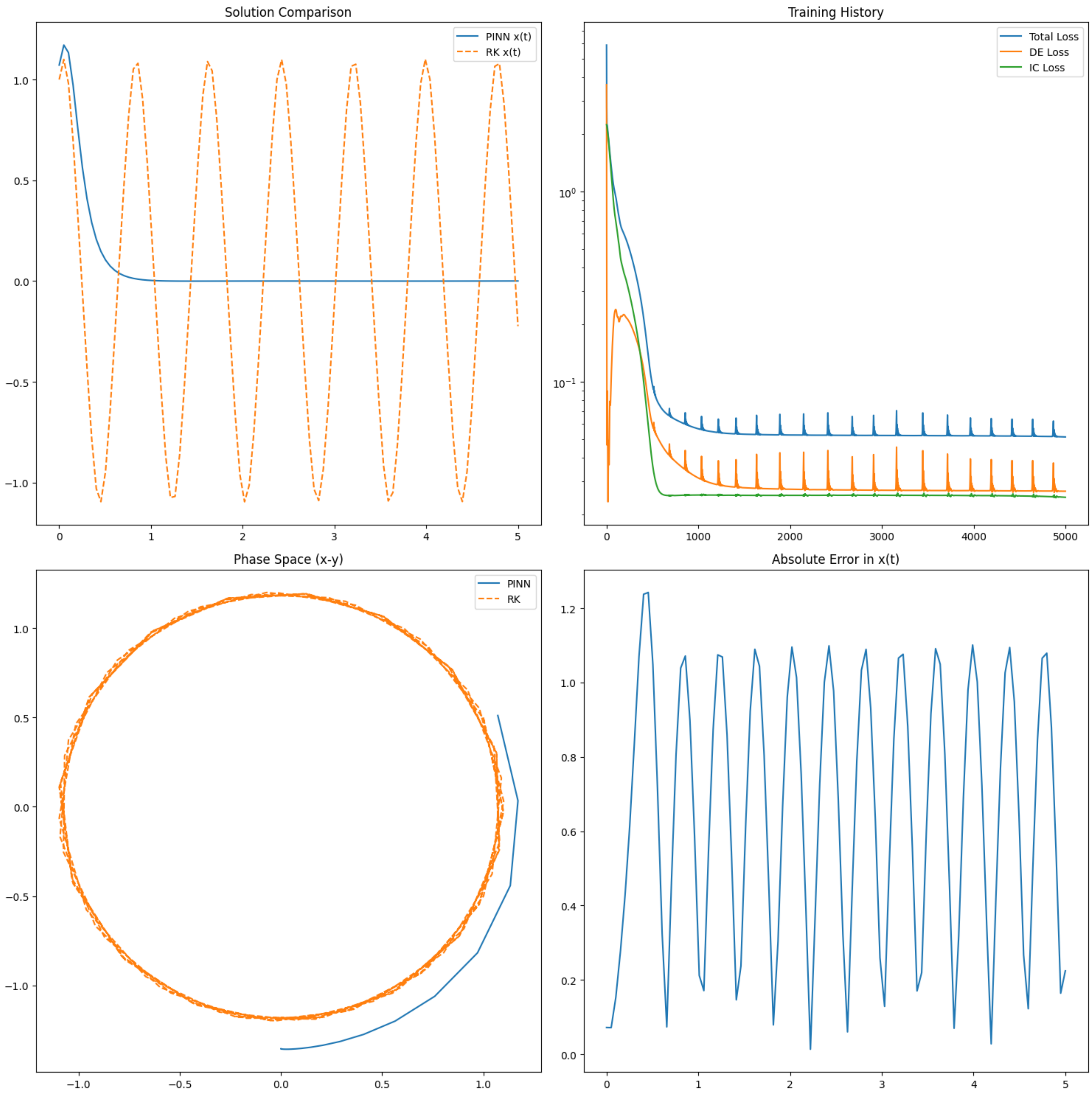
In [ ]:

```
# Time boundaries
end_times = [1, 5, 10, 20]

for tfinal in end_times:
    print(f'\nTraining Lorenz system for tf = {tfinal}')
    de_points = defineCollocationPoints([0, tfinal], 100)
    model = build_model(output_dim=3)

    history = train_model(de_points, model, compute_lorenz_loss)

    # Compare with solve_ivp
    lorenz_exact = solve_ivp(lambda t, u: [
        k * l * ((1/k**2 + l**2) - (1/k**2)) * u[1] * u[2],
        k * l * ((1/l**2) - (1/k**2 + l**2)) * u[0] * u[2],
        (k * l**2) * ((1/k**2) - (1/l**2)) * u[0] * u[1]
    ], [0, tfinal], [x0, y0, z0], t_eval=np.linspace(0, tfinal, 100))

    # Plot results
    fig, axes = plt.subplots(2, 2, figsize=(15, 15))
    t_eval = lorenz_exact.t.reshape(-1, 1)
    predictions = model.predict(t_eval)

    # Solution comparison
    axes[0,0].plot(t_eval, predictions[:,0], label='PINN x(t)')
    axes[0,0].plot(t_eval, lorenz_exact.y[0], '--', label='RK x(t)')
    axes[0,0].set_title('Solution Comparison')
    axes[0,0].legend()

    # Loss history
    axes[0,1].semilogy(history['loss'], label='Total Loss')
    axes[0,1].semilogy(history['de_loss'], label='DE Loss')
    axes[0,1].semilogy(history['iv_loss'], label='IC Loss')
    axes[0,1].set_title('Training History')
    axes[0,1].legend()

    # Phase space
    axes[1,0].plot(predictions[:,0], predictions[:,1], label='PINN')
    axes[1,0].plot(lorenz_exact.y[0], lorenz_exact.y[1], '--', label='RK')
    axes[1,0].set_title('Phase Space (x-y)')
    axes[1,0].legend()

    # Error
    axes[1,1].plot(t_eval, np.abs(predictions[:,0] - lorenz_exact.y[0]))
    axes[1,1].set_title('Absolute Error in x(t)')

    plt.tight_layout()
    plt.show()
```

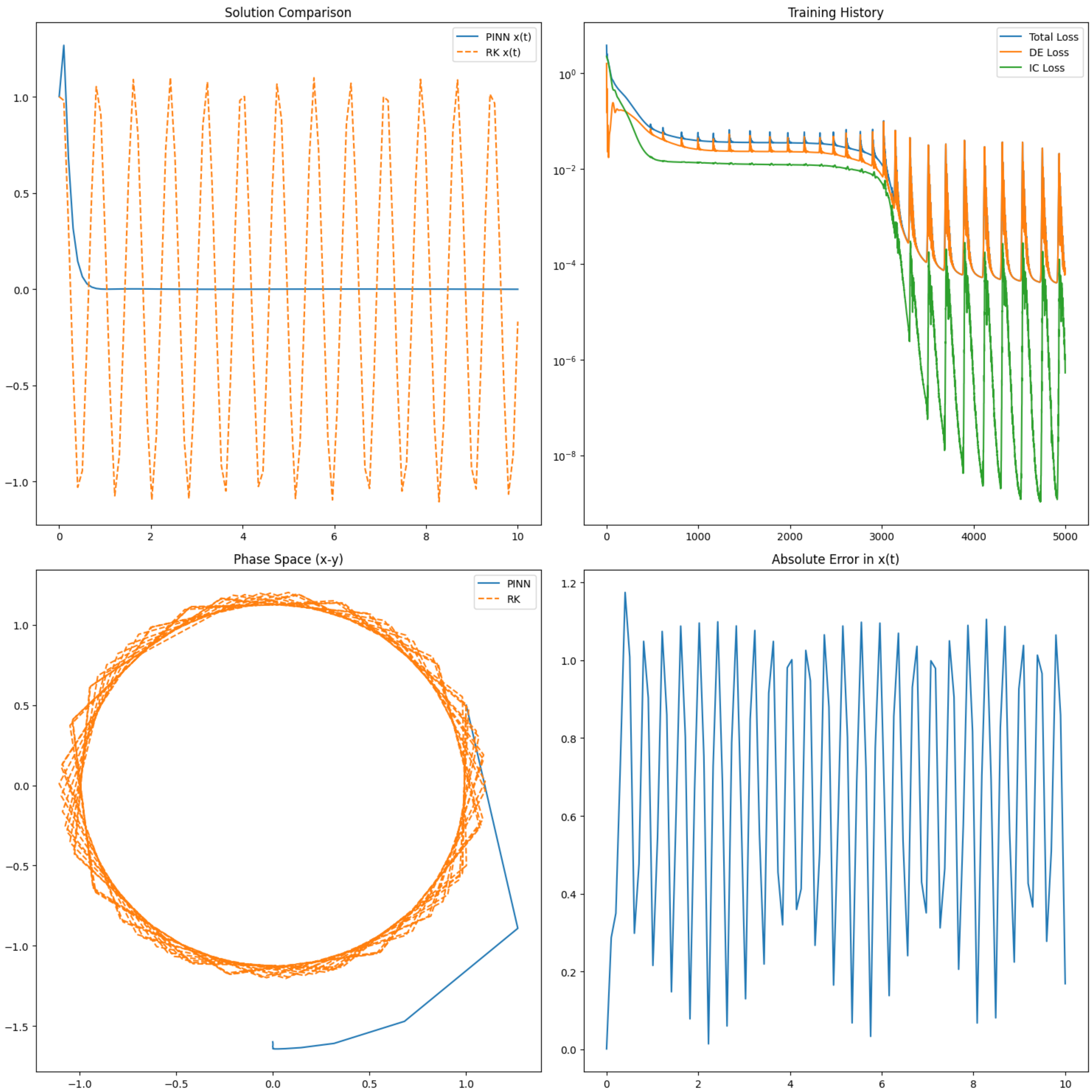
```
Training Lorenz system for tf = 1
Epoch 0: Loss = 2.6625e+00
Epoch 500: Loss = 9.0233e-02
Epoch 1000: Loss = 8.9062e-02
Epoch 1500: Loss = 8.8887e-02
Epoch 2000: Loss = 9.8772e-02
Epoch 2500: Loss = 8.8829e-02
Epoch 3000: Loss = 8.8821e-02
Epoch 3500: Loss = 8.8834e-02
Epoch 4000: Loss = 8.8810e-02
Epoch 4500: Loss = 8.8807e-02
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step
```



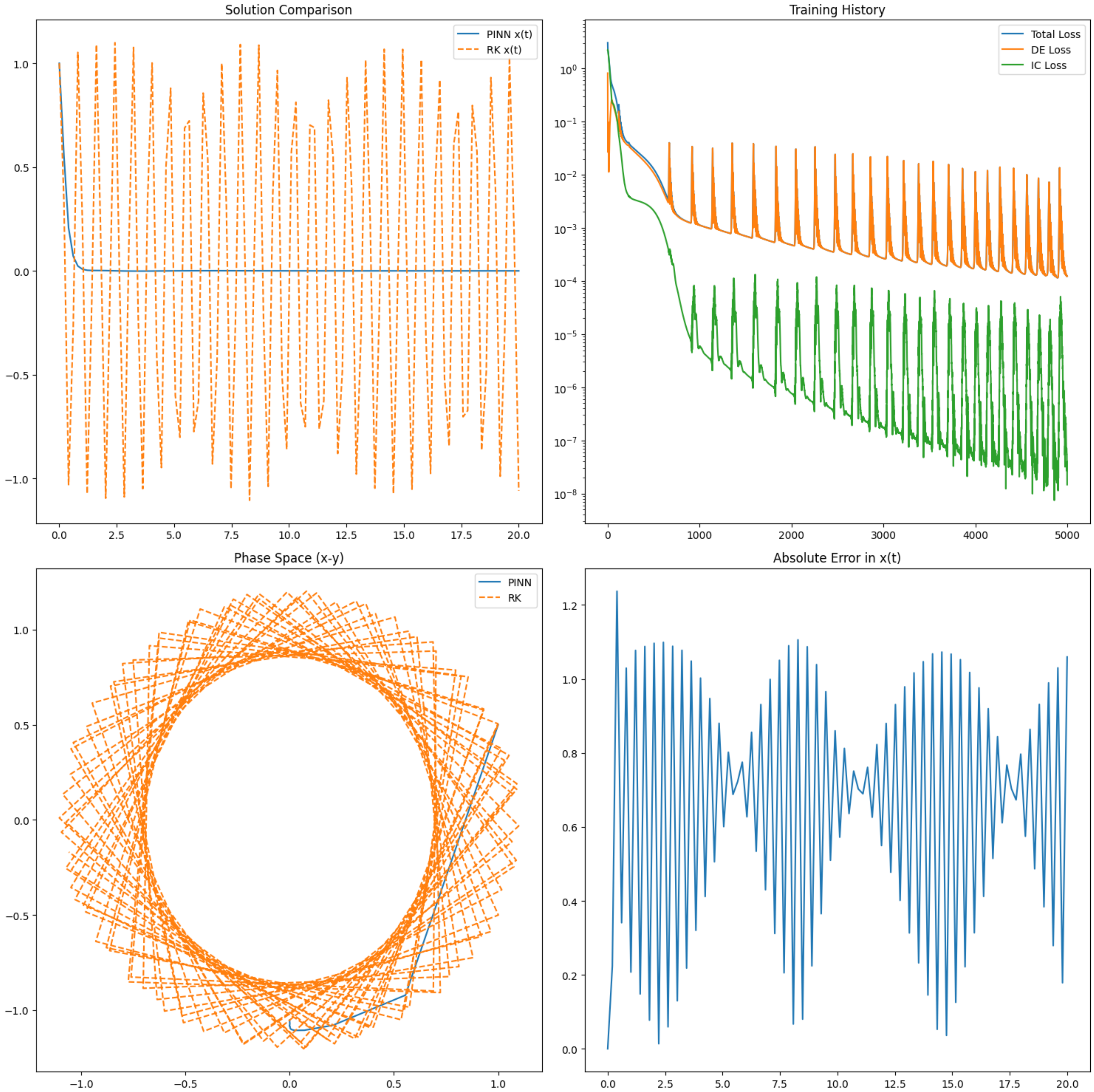
```
Training Lorenz system for tf = 5
Epoch 0: Loss = 5.9002e+00
Epoch 500: Loss = 9.5985e-02
Epoch 1000: Loss = 5.6591e-02
Epoch 1500: Loss = 5.3006e-02
Epoch 2000: Loss = 5.2508e-02
Epoch 2500: Loss = 5.2318e-02
Epoch 3000: Loss = 5.2198e-02
Epoch 3500: Loss = 5.2102e-02
Epoch 4000: Loss = 5.2274e-02
Epoch 4500: Loss = 5.1790e-02
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step
```



```
Training Lorenz system for tf = 10
Epoch 0: Loss = 3.8544e+00
Epoch 500: Loss = 6.8709e-02
Epoch 1000: Loss = 4.0299e-02
Epoch 1500: Loss = 3.5587e-02
Epoch 2000: Loss = 3.6326e-02
Epoch 2500: Loss = 3.1251e-02
Epoch 3000: Loss = 1.2235e-02
Epoch 3500: Loss = 1.6475e-03
Epoch 4000: Loss = 7.1121e-05
Epoch 4500: Loss = 4.4952e-05
WARNING:tensorflow:5 out of the last 9 calls to <function TensorFlowTrainer.make_predict_function.<locals>.one_step
_on_data_distributed at 0x78b5c5485760> triggered tf.function retracing. Tracing is expensive and the excessive num
ber of tracings could be due to (1) creating @tf.function repeatedly in a loop, (2) passing tensors with different
shapes, (3) passing Python objects instead of tensors. For (1), please define your @tf.function outside of the loop
. For (2), @tf.function has reduce_retracing=True option that can avoid unnecessary retracing. For (3), please refe
r to https://www.tensorflow.org/guide/function#controlling_retracing and https://www.tensorflow.org/api_docs/pytho
n/tf/function for  more details.
1/4 ━━━━━━━━━━━━━━━━━━━━ 0s 86ms/step
WARNING:tensorflow:6 out of the last 12 calls to <function TensorFlowTrainer.make_predict_function.<locals>.one_ste
p_on_data_distributed at 0x78b5c5485760> triggered tf.function retracing. Tracing is expensive and the excessive nu
mber of tracings could be due to (1) creating @tf.function repeatedly in a loop, (2) passing tensors with different
shapes, (3) passing Python objects instead of tensors. For (1), please define your @tf.function outside of the loop
. For (2), @tf.function has reduce_retracing=True option that can avoid unnecessary retracing. For (3), please refe
r to https://www.tensorflow.org/guide/function#controlling_retracing and https://www.tensorflow.org/api_docs/pytho
n/tf/function for  more details.
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
```



```
Training Lorenz system for tf = 20
Epoch 0: Loss = 3.0706e+00
Epoch 500: Loss = 1.2897e-02
Epoch 1000: Loss = 1.1327e-03
Epoch 1500: Loss = 7.0087e-04
Epoch 2000: Loss = 4.7588e-04
Epoch 2500: Loss = 1.5518e-03
Epoch 3000: Loss = 2.6710e-04
Epoch 3500: Loss = 2.1201e-04
Epoch 4000: Loss = 1.1444e-02
Epoch 4500: Loss = 1.4728e-04
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step
```


Solution Comparison
PINN x(t)
RK x(t)
1.0
0.5
0.0
−0.5
−1.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Training History
Total Loss
DE Loss
IC Loss
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
0
1000
2000
3000
4000
5000
Phase Space (x-y)
PINN
RK
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0
Absolute Error in x(t)
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0