Numerical Bonus Report 2

Objective:

Fit the mathematical model
$f(x; a_0, a_1) = a_0 (1 - e^{(-a_1 x)})$
to a set of data using Python programming.

Dataset:

x	0.25	0.75	1.25	1.75	2.25
y	0.28	0.57	0.68	0.74	0.79

Instructions:

- Code the fitting procedure in Python and submit the code.
- Try each of the following initial guesses for (a_0, a_1): (1, 1), (0.5, 0.5), (−1, 1), (1, −1)
- Use a stopping rule based on either:
  - The sum of squared residuals < 10^{-4}
  - Change in parameters ε_k < 0.01 for k = 0, 1

Required Tasks:

- For each iteration, calculate and record the sum of squared residuals.
- Track and plot the parameter updates ε_0 and ε_1 over iterations.
- Generate plots comparing the model $f(x; a_0, a_1)$ to the actual observations y_i.

Python Code Template:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit

# Data
x_data = np.array([0.25, 0.75, 1.25, 1.75, 2.25])
y_data = np.array([0.28, 0.57, 0.68, 0.74, 0.79])

# Model function
def model(x, a0, a1):
    return a0 * (1 - np.exp(-a1 * x))

# Initial guesses
guesses = [(1, 1), (0.5, 0.5), (-1, 1), (1, -1)]

for i, guess in enumerate(guesses):
    print(f"\nInitial Guess {i+1}: a0={guess[0]}, a1={guess[1]}")
```

```
popt, _ = curve_fit(model, x_data, y_data, p0=guess, maxfev=10000)
y_pred = model(x_data, *popt)
residuals = y_data - y_pred
ss_res = np.sum(residuals**2)
print("Fitted Parameters:", popt)
print("Sum of Squared Residuals:", ss_res)
plt.figure()
plt.plot(x_data, y_data, 'o', label='Observed')
plt.plot(x_data, y_pred, '-', label='Fitted')
plt.title(f'Fit using initial guess {guess}')
plt.legend()
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.show()
```

Expected Output:

- Numerical values of a_0, a_1 after fitting.
- Plots for each set of initial guesses.
- Graphs of residual sums and parameter change over iterations.

Note: AI is used to write this document in a well oriented format.