oblig_1b/terningkast_2.py

```python
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Importing csv files
p1 = pd.read_csv("oblig_1b/Terning_20.csv")
# Function to perform linear regression, plot, calculate SSE, and calculate SE
def linear_regression_and_plot(df, x_col, y_col, num_points, file_suffix):
    # Create a new DataFrame with columns "1" and the independent variable
    X = pd.DataFrame({
        "Intercept": np.ones(num_points),
        x_col: df[x_col].iloc[0:num_points]
    })

    # Define y as the dependent variable
    y = df[y_col].iloc[0:num_points]

    # Convert the DataFrames to numpy arrays
    X_np = X.values
    y_np = y.values.reshape(-1, 1)

    # Calculating regression line
    # Calculate the dot product of the matrix and its transpose
    XtX = np.dot(X_np.T, X_np)
    XtX_inv = np.linalg.inv(XtX)
    Xty = np.dot(X_np.T, y_np)

    # Calculate beta coefficients
    beta = np.dot(XtX_inv, Xty)
    alpha = beta[0]
    print(f'alpha ({file_suffix}): {alpha}')
    print(f'beta ({file_suffix}): {beta[1]}')

    # Calculate regression line and residuals
    regression_line = np.dot(X_np, beta)
    residuals = y_np - regression_line

    # Calculate SSE (Sum of Squared Errors)
    SSE = np.sum(residuals**2)
    print(f'SSE ({file_suffix}): {SSE}')

    # Calculate SE (Standard Error of the regression)
    # Degrees of freedom: num_points - number of parameters
    dof = num_points - len(beta)
    SE = np.sqrt(SSE / dof)
    print(f'SE ({file_suffix}): {SE}')

    # Plotting the scatter plot and regression line
    plt.scatter(df[x_col][0:num_points], df[y_col][0:num_points], color="pink", marker="o")
    plt.plot(X_np[:, 1], regression_line, color="blue", linewidth=3)
    plt.xlabel(x_col)
    plt.ylabel(y_col)
    plt.title(f'Scatter Plot with Regression Line {file_suffix}')
    plt.savefig(f"oblig_1b/plot_{file_suffix}.png")
    plt.show()

```

```
57      return alpha, beta[1], SSE, SE
58
59  # Perform linear regression, plot, calculate SSE, and calculate SE for the first
    subset
60  alpha_5, beta_5, SSE_5, SE_5 = linear_regression_and_plot(p1, "Hoyde", "Lengde", 5,
    "5_points")
61
62  # Perform linear regression, plot, calculate SSE, and calculate SE for the all data
    subset
63  alpha_30, beta_30, SSE_30, SE_30 = linear_regression_and_plot(p1, "Hoyde", "Lengde"
    , 30, "30_points")
64
```