```python
import subprocess
import sys

# Function to install packages if missing
def install_if_missing(packages):
    for package in packages:
        try:
            __import__(package)
        except ImportError:
            print(f"Installing missing package: {package}")
            subprocess.check_call([sys.executable, "-m", "pip", "install", package])

# List of required packages
required_packages = [
    "pandas",
    "numpy",
    "matplotlib",
    "scipy",
    "statsmodels",
    "pymannkendall"
]

# Install missing packages
install_if_missing(required_packages)

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import linregress
from statsmodels.tsa.seasonal import STL
from statsmodels.nonparametric.smoothers_lowess import lowess
import pymannkendall as mk
from pymannkendall import sens_slope
import os

def analyze_trends(file_path, x_label, y_label, output_csv="trend_results.csv"):
    """
    Performs STL, LOESS, and Mann-Kendall trend analysis.
    Computes Sen's Slope, overlays a linear fit, and saves trend statistics & processed data to CSV.
    Saves the plot as PLOT_{filename}.png with predefined axis labels.

    Parameters:
    - file_path (str): Path to the input CSV file.
    - x_label (str): Label for the x-axis.
    - y_label (str): Label for the y-axis.
    - output_csv (str): Filename of the CSV where summary results will be saved.
    """

    # Load data and ensure correct headers
    df = pd.read_csv(file_path)
    df.columns = df.columns.str.strip()  # Strip accidental spaces

    # Detect first column (date) and second column (numeric value)
    date_col = df.columns[0]
    value_col = df.columns[1]

    # Convert date column to proper datetime format (end of month)
    df[date_col] = pd.to_datetime(df[date_col], format="%Y-%m", errors="coerce") +
pd.offsets.MonthEnd(1)
    df.set_index(date_col, inplace=True)

    # STL Decomposition
    stl = STL(df[value_col], seasonal=13)
    result = stl.fit()
    df["STL_Trend"] = result.trend

    # LOESS smoothing
    df["LOESS_Trend"] = lowess(df["STL_Trend"].dropna(),
np.arange(len(df["STL_Trend"].dropna())), frac=0.1)[:, 1]

    # Mann-Kendall trend test
    mk_result = mk.original_test(df["STL_Trend"].dropna(), alpha=0.05)

    # Compute trend statistics
    p_value = mk_result.p
    z_score = mk_result.z
    test_statistic = mk_result.s
    slope_result = sens_slope(df["STL_Trend"].dropna())

    # Linear Regression Fit
    slope, intercept, _, _, _ = linregress(df.index.year, df["STL_Trend"])
    df["Linear_Trend"] = intercept + slope * df.index.year

    # Print test results
    print(f"Full Mann-Kendall p-value: {p_value}")
    print(f"Mann-Kendall Z-score: {z_score}")
    print(f"Mann-Kendall Test Statistic (S-value): {test_statistic}")
    print(f"Sen's Slope: {slope_result.slope:.4f} per month")

    # Save summary results to CSV
    filename = os.path.basename(file_path)
    results_df = pd.DataFrame([[filename, p_value, z_score, test_statistic,
slope_result.slope]],
                              columns=["Filename", "p-value", "Z-score", "Test Statistic", "Sen's Slope"])

    if not os.path.exists(output_csv):
        results_df.to_csv(output_csv, index=False)
    else:
        results_df.to_csv(output_csv, mode='a', header=False, index=False)

    print(f"Summary results saved to {output_csv}")

    # Save processed regression values to a new CSV
    reg_values_filename = f"REG_VALUES_{filename}"
    df_out = df[[value_col, "STL_Trend", "LOESS_Trend", "Linear_Trend"]].reset_index()
    df_out.columns = ["Year-Month", "Original Data", "STL Data", "LOESS Data", "Linear
Trend Data"]
    df_out.to_csv(reg_values_filename, index=False)

    print(f"Processed regression values saved to {reg_values_filename}")

    # Save the plot
    plot_filename = f"PLOT_{filename}.png"

    # Plotting
    plt.figure(figsize=(12, 6))
    plt.plot(df.index, df[value_col], label="Original Data", alpha=0.5)
    plt.plot(df.index, df["STL_Trend"], label="STL Trend", color="orange", linewidth=2)
    plt.plot(df.index, df["LOESS_Trend"], label="LOESS Smoothed Trend", color="green",
linewidth=2)
    plt.plot(df.index, df["Linear_Trend"], label="Linear Fit", color="red",
linestyle="dashed")

    # Add an inset box for Mann-Kendall results (including S-value)
    textstr = (f"Mann-Kendall Test:\nTrend: {mk_result.trend}\n"
               f"p-value: {p_value:.3f}\n"
               f"Z-score: {z_score:.3f}\n"
               f"Test Statistic: {test_statistic}\n"
               f"Sen's Slope: {slope_result.slope:.4f}/month")

    props = dict(boxstyle="round", facecolor="white", alpha=0.7)
    plt.gca().text(0.02, 0.98, textstr, transform=plt.gca().transAxes,
                   fontsize=10, verticalalignment="top", bbox=props)

    plt.legend()
    plt.title(f"{value_col} Trend Analysis")
    plt.xlabel(x_label)
    plt.ylabel(y_label)

    # Save figure and show plot
    plt.savefig(plot_filename, dpi=300, bbox_inches="tight")
    plt.show()

    print(f"Plot saved as {plot_filename}")

# Example usage
file_path = "MONTHLY_PROC_Metheringham_Fen_2000_2024.csv"  # Replace with required filename
x_label = "Year-Month"  # Set x-axis label as a variable
y_label = "Groundwater Storage (mm)"  # Set y-axis label as a variable

analyze_trends(file_path, x_label, y_label)
```