Q2 - Unevenly Sampled Time Series Analysis

Implementation

The question is to perform Unevenly Sampled Time Series Analysis using Lomb Scale Periodogram and ARIMA model. Here, Lomb Scale Periodogram is implemented from scratch and ARIMA is implemented using the builtin routine of MATLAB. The `matlab\Q2` folder has all the MATLAB files related to this question.

Lomb Scale Periodogram

```
%% Lomb Scale Periodogram using Gradient Descent
function [frequencies, powers, cost_history, a, b] = lomb_scale_periodogram(time, signal,
    %% Input Arguments
    %   time: Time instances of the signal
    %   signal: Signal sampled at the instants specified in time
    %   alpha: Learning rate (scalar)
    %   epochs: Number of epochs (scalar)

    %% Output Arguments
    %   Returns the power spectral density estimate
    %       frequencies: List of frequncies
    %       powers: Powers at each frequency
    %   cost_history: Cost at every epoch using the LS method
    %   LS Periodogram Estimates
    %       a: Coefficients of cosine term
    %       b: Coefficients of sine term

    %% Initialize variables
    N = length(time); % Length of the signal
    if ~isempty(varargin)
        K = varargin{1}; % User-defined Resolution of the frequency range considered for
    else
        K = 1000; % Resolution of the frequency range considered for analysis
    end
    % Compute the frequency range for analysis
    lower_bound = 0; % Lower bound
    upper_bound = floor((N/time(end))/2); % Upper bound (Fs/2)
    frequencies = lower_bound:(upper_bound/K):upper_bound; % Construct frequency range
    m = length(frequencies); % Number of frequencies in the analysis range
    powers = zeros(m, 1); % Power of each signal in the frequency range
    a = zeros(m, 1); % Coefficient for cosine term
    b = zeros(m, 1); % Coefficient for sine term
    cost_history = zeros(epochs, 1); % Cost history
    %% Pre-process signal
    missing_indices = isnan(signal); % Identify the indices of missing data
    non_missing_signal = signal(~missing_indices); % Construct the signal omitting the mi
    mean_value = mean(non_missing_signal); % Compute mean
    centered_non_missing_signal = non_missing_signal - mean_value; % Mean center the sigr
    % Processed signal
    y = signal;
    y(~missing_indices) = centered_non_missing_signal; %For non-missing indice,use the me
    y(missing_indices) = mean_value; % For missing indices, use the mean value

    %% Gradient Descent
    for epoch = 1:epochs
        % Compute y_hat and cost
        y_hat = zeros(N, 1);
        for i = 1:m
           w = 2*pi*frequencies(i); % w=2?f
           y_hat = y_hat + (a(i)*cos(w*time) + b(i)*sin(w*time)); % construct y_hat
        end
        residual = y-y_hat; % Residual
        cost_history(epoch) = (sum(residual.^2)/2*N); % Oridinary Least Squares

        % Compute gradients
        for i = 1:m
            w = 2*pi*frequencies(i);
            grad_a = -sum(residual.*cos(w*time))/N; % Gradient for a
            grad_b = -sum(residual.*sin(w*time))/N; % Gradient for b

            % Parameter updates
            a(i) = a(i) - alpha*grad_a;
            b(i) = b(i) - alpha*grad_b;
        end
    end

    %% Compute powers of each signal frequency
    for i = 1:m
        w = 2*pi*frequencies(i);
        s = a(i)*cos(w*time) + b(i)*sin(w*time);
        powers(i) = sum(s.^2) / N;
    end
end
```

The above implementation uses Gradient Descent for optimization. The number of frequencies to search for is inferred from the data (analysing the sample frequency of the data). The number of divisions between the range can be specified. The higher the number of divisions, the better the output. Other helper functions used in the above analysis can be found in the `matlab\Q2` folder.

Synthetic Data

As per the question, synthetic data is generated as a summation of two sine waves. Lomb Scale Periodogram is applied and the predictions are made. Here is the summary of the analysis,

Metric	Original Signal vs Reconstructed Signal	Noisy Signal vs Reconstructed Signal
NMSE	0.61553	0.0027936
MAPE	95.218	4.6254

From the above table, we can infer that while the Lomb Scale Periodogram works well in cleaning up noise and missing values (as indicated by the low NMSE and MAPE for the noisy signal), it may not be as effective in accurately replicating the original signal (as indicated by the high NMSE and MAPE for the original signal).

Real Dataset - Tesla Stocks

For the real dataset, Tesla stock price from 2011 to 2020 is used. The first 80% data is used for training and remaining 20% is used for testing. Data is mean centered before computing the periodogram. Both ARIMA and Lomb Scale Periodogram are used to predict the test data in this question. The following is the analysis results:

Metric	LSP	ARIMA
NMSE	1.3658	1.2682
MAPE	20.479	22.55

From the above data we can infer that,

- The NMSE values for both models are **relatively close**, indicating that both LSP and ARIMA perform similarly in terms of squared error between predicted and actual stock prices. Lower NMSE values indicate better performance in terms of minimizing squared errors.

- The MAPE values for LSP and ARIMA differ slightly, with **LSP having a lower MAPE**. Lower MAPE values indicate better accuracy in predicting stock prices, as they represent a smaller percentage of error relative to the actual prices.

Both models, LSP and ARIMA, have similar NMSE values, suggesting comparable performance in terms of squared error between predicted and actual stock prices. LSP outperforms ARIMA slightly in terms of MAPE, indicating slightly better accuracy in predicting stock prices with lower percentage error. When considering both NMSE and MAPE, LSP appears to be slightly more accurate than ARIMA in predicting stock prices for the given task.