# Time Series Imputation Via Pattern Matching

by

Weiran Xu

A project submitted to the Graduate Program in

Mathematics & Statistics

in conformity with the requirements

for the Degree of Masters of Science

Queen's University

Kingston, Ontario, Canada

... 2024

# Abstract

This report focuses on a nonparametric pattern matching approach for imputation. For a given gap, a simple imputation method, such as linear interpolation, can be used to fill in the gap. Therefore, for the first part of research we generated artificial time series, simulated missing data under MCAR conditions with varying gap structures, applied different interpolation algorithms, and computed performance metrics. Finallya visualisation tool was used to find the most suitable algorithm. The discovery reflected the high accuracy of HWI algorithm for time series interpolation.

For the second part of research, we created a target segment Combined with available data immediately before and after the gap. We then selected multiple predictive segments, each the same length as the target segment and each containing full data. In this process, we explored similarity measures between predictive segments and the target segment. Next, we combined the predictive segments weighted by their similarity to the target segment to produce a single predictive segment, which will be used to replace the naively imputed values in the target segment. Finally, using R to produce a weighted combination of segments of a time series to produce an imputation of missing values, and evaluate the performance of the imputation. I once iteratively filled in gaps within a time series by identifying and using the most similar segments from the surrounding data to estimate the missing values. This approach is particularly useful in situations where data is missing in a sequential manner instead of simple interpolation methods which may not match the underlying patterns in the

data.

# Acknowledgements

I would like to express my deepest gratitude to Professor Glen Takahara, whose expertise and guidance were invaluable throughout this research project. His support and insightful feedback greatly enriched this work. I also wish to thank the Dr.Sophie Bull and Wesley Burr for providing me with the interpTools package, which the simulation and visualisation functions of time series interpolation have greatly improved my productivity and led me to the final discovery. This report would not have been possible without the contributions of these individuals.

# Table of Contents

# List of Figures

# List of Tables

# 1 Introduction

## 1.1 Summary

- **Time Series**
  - Time series analysis focuses on studying data points collected at specific intervals over time.
  - It aims to model patterns such as trends, cycles, and seasonal variations for predicting future events.
  - The field developed in the early 20th century, with contributions from Norbert Wiener and George Udny Yule.
  - Modern models like ARIMA and GARCH have emerged with advancements in computational power.
  - Time series analysis is widely applied in finance, economics, meteorology, healthcare, engineering, and social sciences.
  - It is crucial for understanding historical data and anticipating future developments.
- **Interpolation**
  - Interpolation addresses the challenge of missing data in time series analysis.

  - Missing data can result from equipment failures, recording errors, or other disruptions.
  - Interpolation estimates missing data points based on surrounding data, maintaining the integrity of the time series.
  - Accurate imputation of missing values is essential for reliable analysis in fields like economics, finance, and natural sciences.

- **Pattern Matching for Time Series Imputation**
  - Pattern matching is a nonparametric method for interpolating missing data by leveraging existing data structures.
  - It involves identifying predictive segments in the time series that resemble the gap region.
  - By comparing these segments, the method refines the imputation, enhancing accuracy.
  - This approach is particularly useful when simple linear or parametric methods are inadequate.
  - Pattern matching provides a nuanced, context-aware estimation, making it a powerful tool for time series imputation.

## 1.2 Time Series

Time series analysis is a critical and well-established area in the field of statistics and data science, focusing on the examination and understanding of data points collected or recorded at specific intervals over time. Academically, time series refers to the sequential ordering of data points, where each point is associated with a specific time period, enabling the study of temporal dynamics and trends. The core objective is to model the underlying patterns in the data—such as trends, cycles, and seasonal variations—to make informed predictions and insights about future behaviors or events.

The development of time series analysis can be traced back to the early 20th century, when it began to gain prominence as a rigorous mathematical and statistical discipline. Influential figures such as Norbert Wiener, who contributed to the development of the mathematical theory of stochastic processes, and George Udny Yule, who advanced autoregressive models, laid the groundwork for modern time series analysis in [7]. In the ensuing decades, the field expanded, incorporating advances in computational power and methodologies, leading to the development of sophisticated models like ARIMA (AutoRegressive Integrated Moving Average) and GARCH (Generalized Autoregressive Conditional Heteroskedasticity) [7].

The practical applications of time series analysis are vast and deeply embedded in numerous real-life domains. In finance, for example, time series models are used to forecast stock prices, interest rates, and market indices, providing critical insights for investors and policymakers. In the field of economics, time series analysis helps in understanding macroeconomic indicators such as GDP, inflation, and unemployment rates, enabling better economic planning and decision-making in [5].

Beyond finance and economics, time series analysis is essential in fields like meteorology for weather forecasting, in healthcare for monitoring patient vitals over time, and in engineering for analyzing signals in systems and processes. Additionally, in the realm of social sciences, time series data is utilized to track changes in social indicators, such as crime rates or public opinion, over extended periods.

The academic rigor and real-world applicability of time series analysis underscore its importance as a tool not only for understanding historical data but also for anticipating future developments, making it indispensable in both theoretical and applied research.

## 1.3 Interpolation

In time series analysis, one of the fundamental challenges encountered is the presence of missing data points. These gaps can emerge for a variety of reasons, such as equipment failures, human errors in recording, or other unexpected disruptions in the data collection process. Missing data complicates the analytical process since many statistical methods rely on the assumption of a complete and uninterrupted time series. To address this issue, it is common practice to estimate and replace the missing values—a process known as *data imputation*, or (shorter), *interpolation* in [6].

Interpolation involves the estimation of missing data points within a time series based on the available surrounding data. It allows analysts to maintain the integrity of the time series, enabling the application of standard analytical methods that require a continuous data set. This technique is crucial across various domains where time series data is prevalent, including economics, finance, and the natural sciences. In practice, time series observations often contain gaps, making the accurate imputation of missing values a critical task in ensuring the reliability of subsequent analyses.

## 1.4 Pattern Matching for Time Series Imputation

Among the various methods for interpolating missing data, nonparametric approaches, such as pattern matching, offer a robust solution. The pattern matching method leverages the structure of the existing data to estimate missing values in a more informed manner. For instance, consider a time series with a gap where data is missing for a certain period. A simple imputation method, like linear interpolation, could be applied to initially fill in the gap. This interpolated section, along with the surrounding data immediately before and after the gap, forms what is known as the target segment.

The pattern matching approach involves searching for multiple predictive segments within the time series that are similar in length and fully populated with data. These predictive segments are identified based on their resemblance to the target segment in terms of trends, cycles, or other patterns. By comparing these segments, the method refines the initial imputation, enhancing the accuracy of the estimated values [6].

This approach is particularly advantageous in situations where the missing data cannot be accurately estimated through simple linear or parametric methods alone. By utilizing the inherent patterns within the data, pattern matching provides a more nuanced and context-aware estimation, making it a powerful tool in the imputation of missing time series data.

# 2 Interpolation Approach

In this chapter, we used castels/interpTools package to do systematic testing of interpolation algorithms, starting with the creation of arbitrary time series and creating gappy data. Next, 18 algorithms were performed to interpolate on gappy series in parallel. Finally, specific function was used to calculate perfomance metrics between lists of original and interpolated series, and visualization tools produced the final result of performance of each interpolation algorithm. In our research, the performance parameter we used is root mean squared error (RMSE).

## 2.1 Introduction of Castels/interpTools

[1] InterpTools provides a framework for performing comprehensive analysis on the statistical performance of time series interpolators in a test-environment.The workflow involves the generation of artifical time series, simulation of MCAR observations subject to two key gap structure parameters ('proportion missing' and 'gap width'), application of interpolation algorithm candidates, and subsequent computation of a set of performance metrics. Tools for both singular and comparative data visualization allows practitioners to elucidate the most suitable algorithm for similarly-structured datasets in practice, especially in the context of a changing gap structure.

An example of a detailed and comprehensive analysis facilitated by this package is written in S. Castel's MSc thesis "A Framework for Testing Time Series Interpolators"

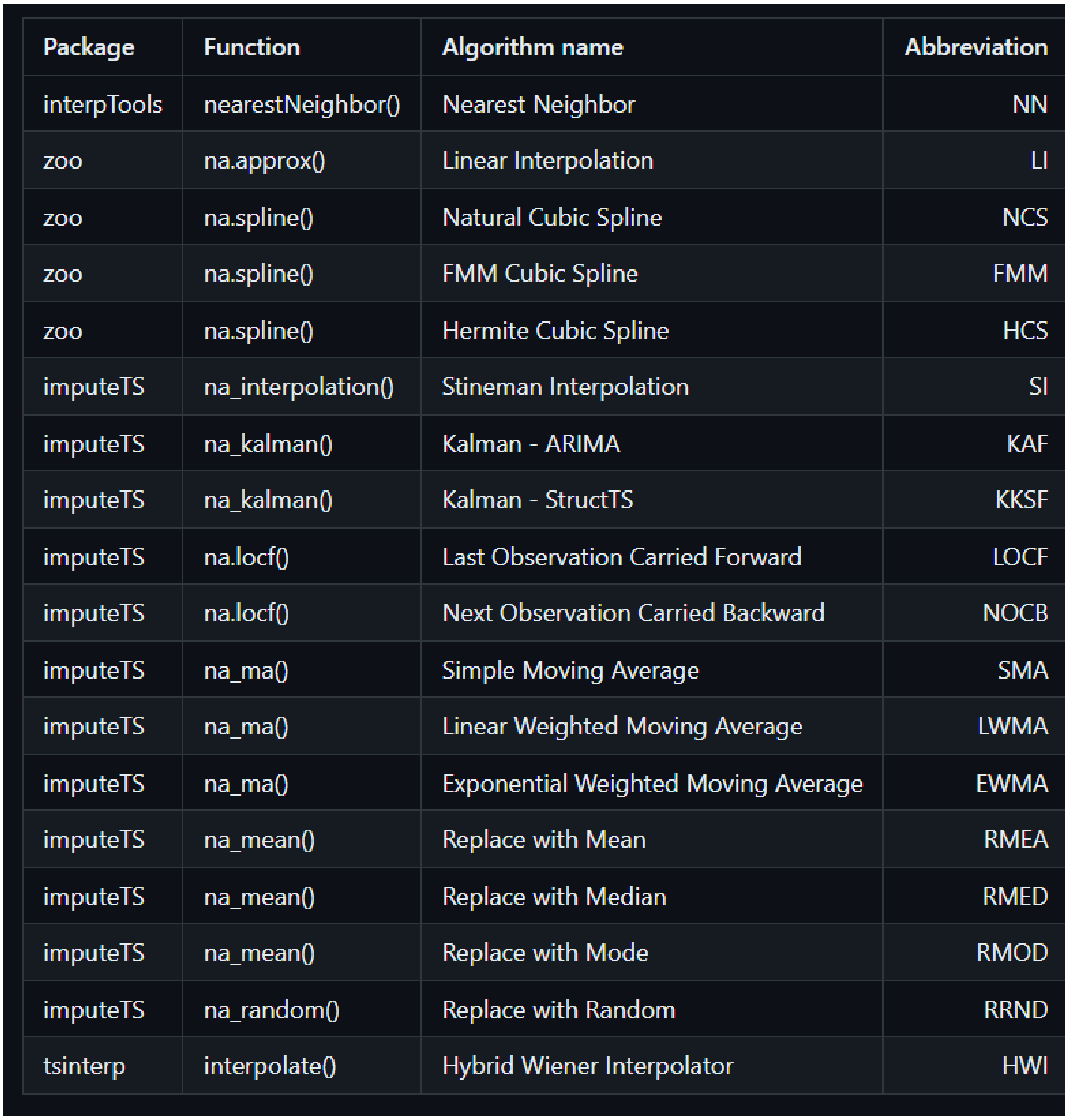
(2020) [3].

Beginning with Simulate time series data with simXt() Data is generated based on the general model of a time series: the addition of a mean, periodic trend, and noise component.

The next step is Generating gaps with $simulateGaps()$.Gap structure is simulated as Missing Completely at Random (MCAR), and defined by two parameters: the proportion of data missing (p), and the gap width (g).We had example: p = [10

Under each possible (p,g) combination, the function will produce K different gap configurations on the original time series.

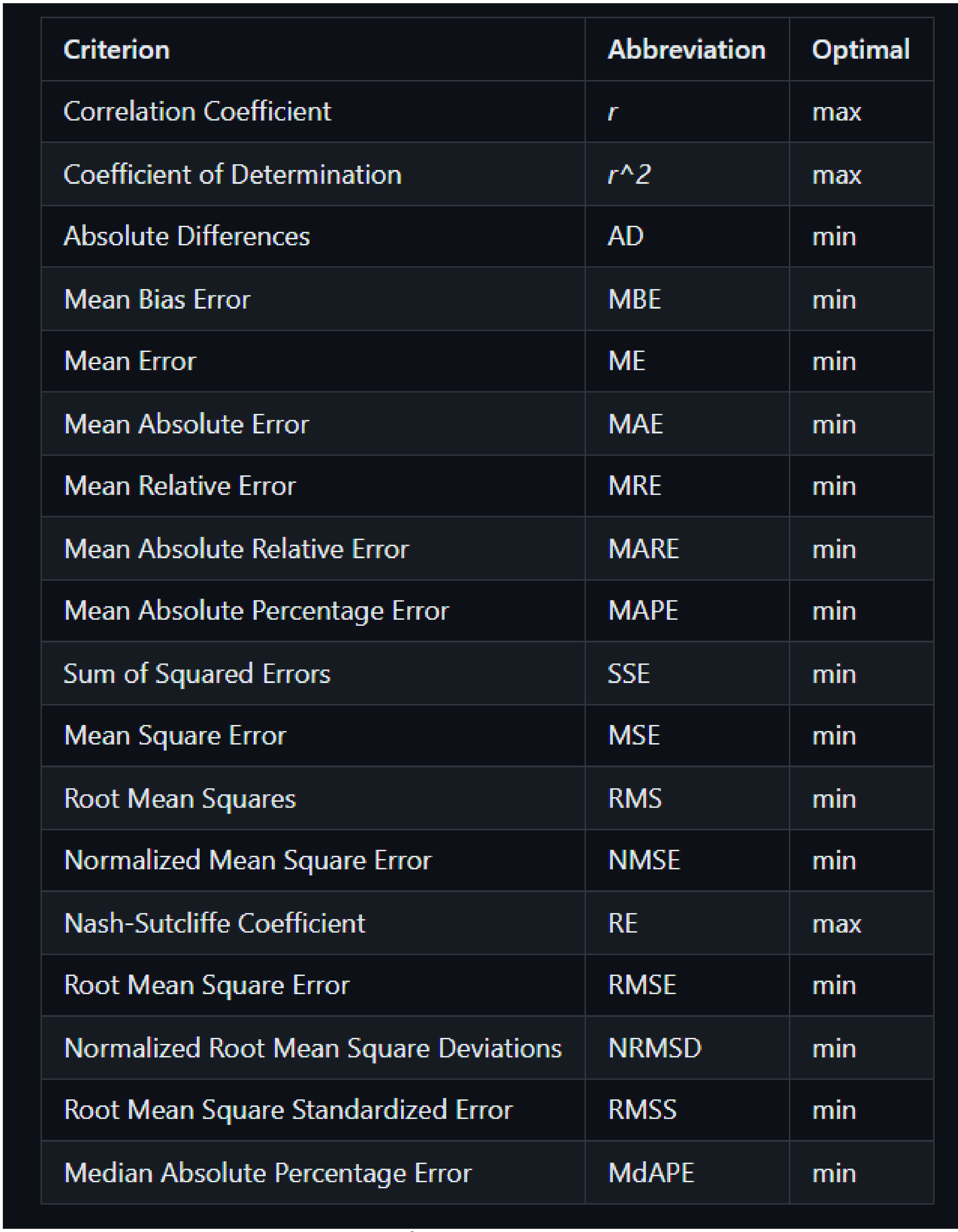
Interpolating the gappy data with $parInterpolate()$. Interpolation is performed on the gappy data, using parallel computing for efficiency. The user can choose from a list of 18 built-in interpolators:



| Package | Function | Algorithm name | Abbreviation |
|---|---|---|---|
| interpTools | nearestNeighbor() | Nearest Neighbor | NN |
| zoo | na.approx() | Linear Interpolation | LI |
| zoo | na.spline() | Natural Cubic Spline | NCS |
| zoo | na.spline() | FMM Cubic Spline | FMM |
| zoo | na.spline() | Hermite Cubic Spline | HCS |
| imputeTS | na_interpolation() | Stineman Interpolation | SI |
| imputeTS | na_kalman() | Kalman - ARIMA | KAF |
| imputeTS | na_kalman() | Kalman - StructTS | KKSF |
| imputeTS | na.locf() | Last Observation Carried Forward | LOCF |
| imputeTS | na.locf() | Next Observation Carried Backward | NOCB |
| imputeTS | na_ma() | Simple Moving Average | SMA |
| imputeTS | na_ma() | Linear Weighted Moving Average | LWMA |
| imputeTS | na_ma() | Exponential Weighted Moving Average | EWMA |
| imputeTS | na_mean() | Replace with Mean | RMEA |
| imputeTS | na_mean() | Replace with Median | RMED |
| imputeTS | na_mean() | Replace with Mode | RMOD |
| imputeTS | na_random() | Replace with Random | RRND |
| tsinterp | interpolate() | Hybrid Wiener Interpolator | HWI |

Figure 2.1: 18 algorithms

Evaluating statistical performance with $performance()$:



| Criterion | Abbreviation | Optimal |
|---|---|---|
| Correlation Coefficient | *r* | max |
| Coefficient of Determination | *r*^2 | max |
| Absolute Differences | AD | min |
| Mean Bias Error | MBE | min |
| Mean Error | ME | min |
| Mean Absolute Error | MAE | min |
| Mean Relative Error | MRE | min |
| Mean Absolute Relative Error | MARE | min |
| Mean Absolute Percentage Error | MAPE | min |
| Sum of Squared Errors | SSE | min |
| Mean Square Error | MSE | min |
| Root Mean Squares | RMS | min |
| Normalized Mean Square Error | NMSE | min |
| Nash-Sutcliffe Coefficient | RE | max |
| Root Mean Square Error | RMSE | min |
| Normalized Root Mean Square Deviations | NRMSD | min |
| Root Mean Square Standardized Error | RMSS | min |
| Median Absolute Percentage Error | MdAPE | min |

Figure 2.2: Performance metrics

Then, we aggregated the performance metrics within each gap specification with $aggregatepf()$. Here, statistical performance is defined as some measure that quantifies the overall degree of deviation between the original values and the interpolated values.

Finally, Visualizing performance as a surface plot with $plotSurface()$. A three-dimensional interactive surface is used to visualize the aggregated performance of an interpolator as the structure of the gappy data changes. Through R's interface, the user can interact with the surface plot widgets by manipulating the camera perspective, adjusting the zoom, and hovering over data points for precise numerical information.

Optimal performance corresponds to an extreme point on the surface; either a maximum or minimum, depending on the definition of optimal. Multiple interpolations can be compared by layering surfaces on top of one another, where the best interpolation for a particular gap structure will be at an extremum at the corresponding (p,g) coordinate point.

## 2.2 Steps of Systematic Analysis

After installing and library required packages, we defined four arbitrary time series. All of them had 1000 data points, which is $N = 1000$ and with different degree of the polynomial in the mean component, number of sinusoids to generate in the trend component and sample frequencies for the trend component, etc.

Next, we combine four time series into a list and create gappy data with $p = (0.05, 0.10, 0.15, 0.20)$, $g = (1, 5, 10)$ and $K = 10$, and execute 16 algorithms to interpolate the gappy data. Unforunately, we failed to run the code with algorithm natural cubic spline and cubic spline, due to unknown system error. We separated 16 algorithms into 7 kinds and used $parInterpolate()$ to fill the gap and completed aggregation. Each group contains different but similar algorithms. As a result, we got 3-D plots of each group performance on filling gappy series, reflected as RMSE.

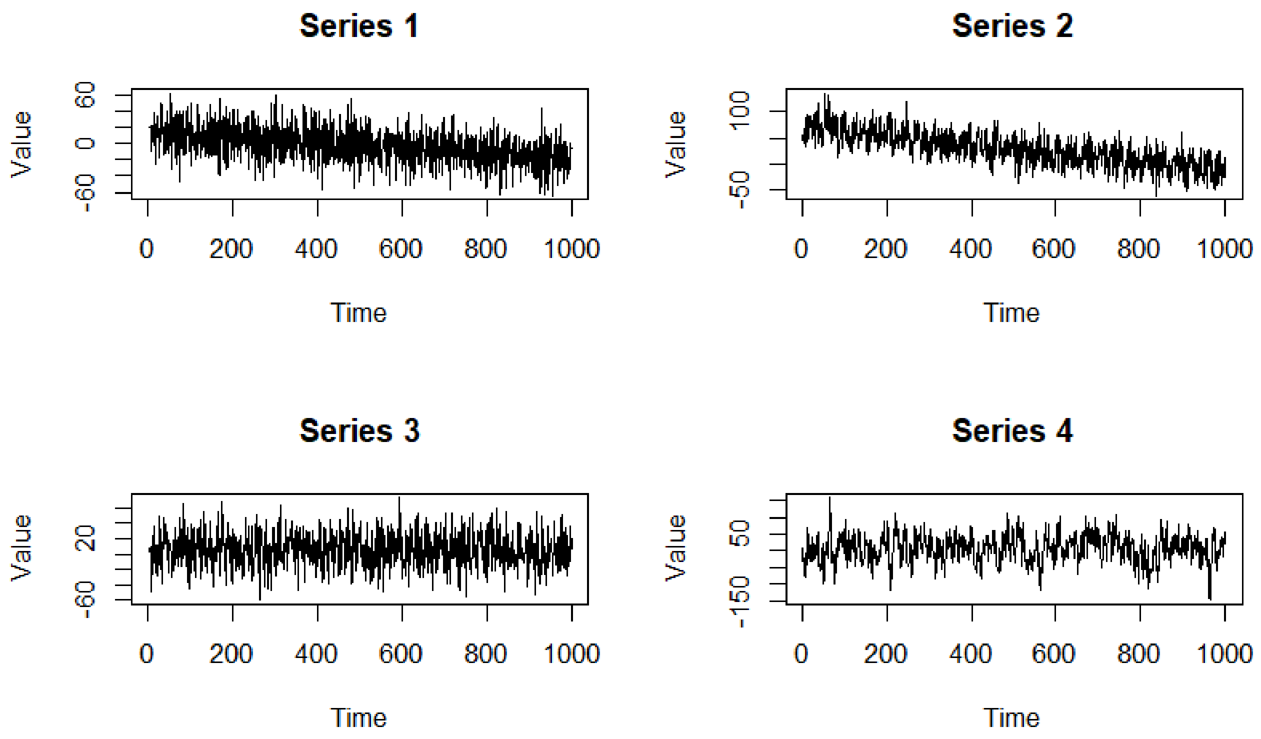


Figure 2.3: Four time series with length 1000

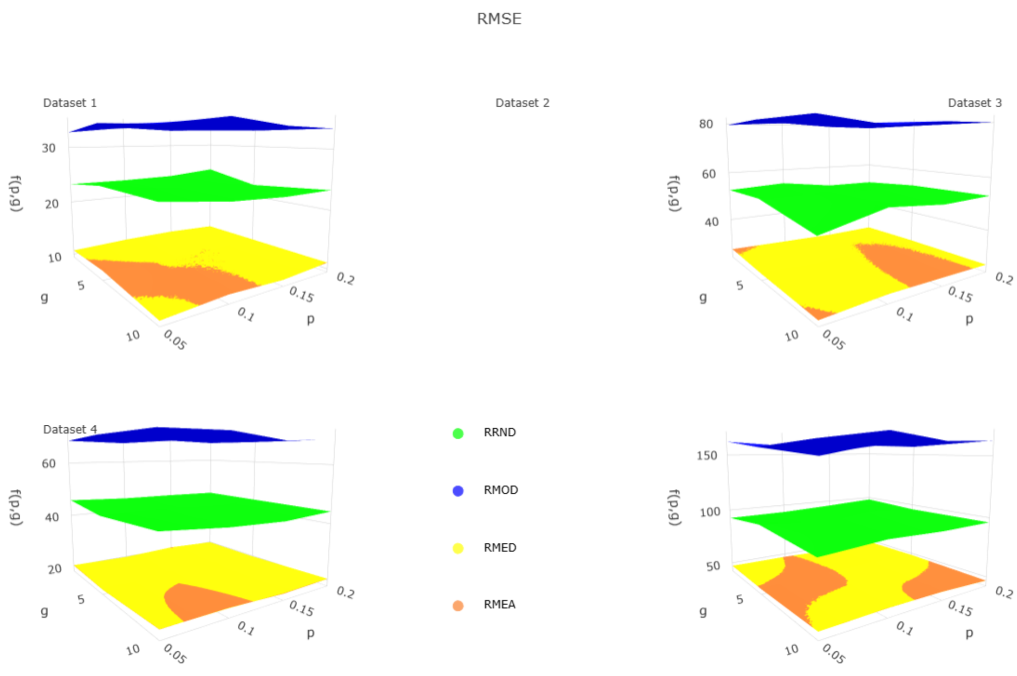


Figure 2.4: RMED,RMOD,RMEA,RRND

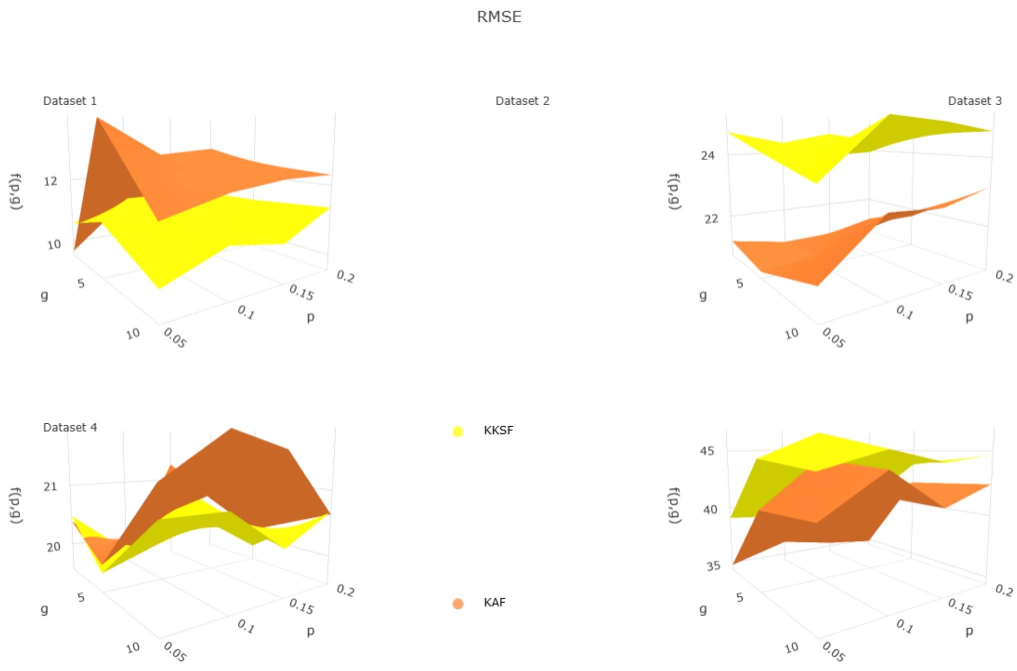


Figure 2.5: KKSF,KAF

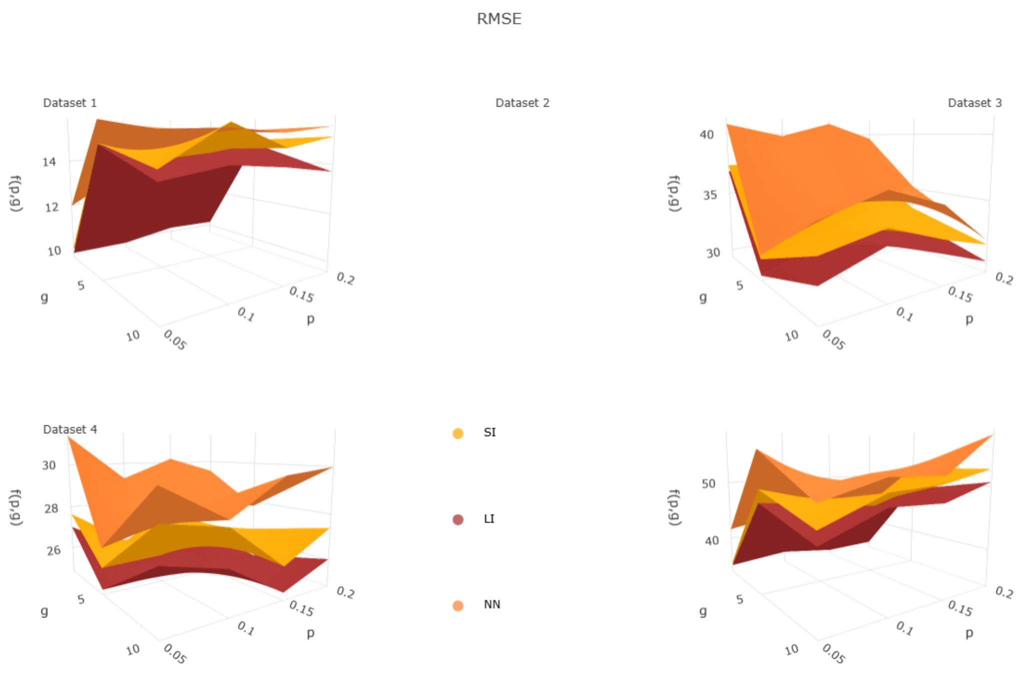


Figure 2.6: SI,LI,NN

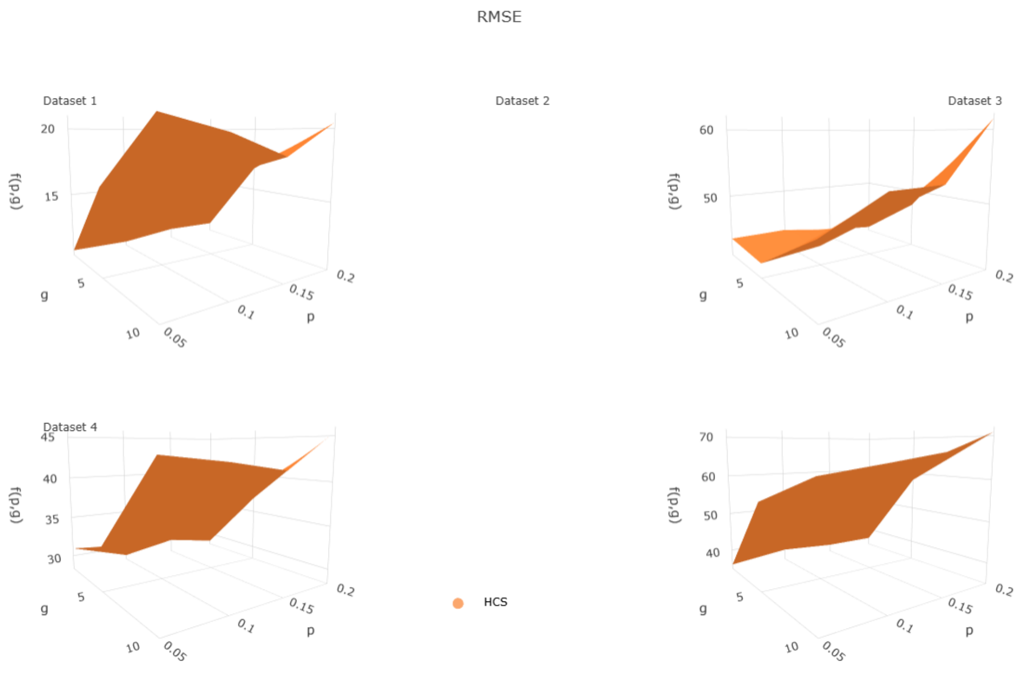


Figure 2.7: HCS

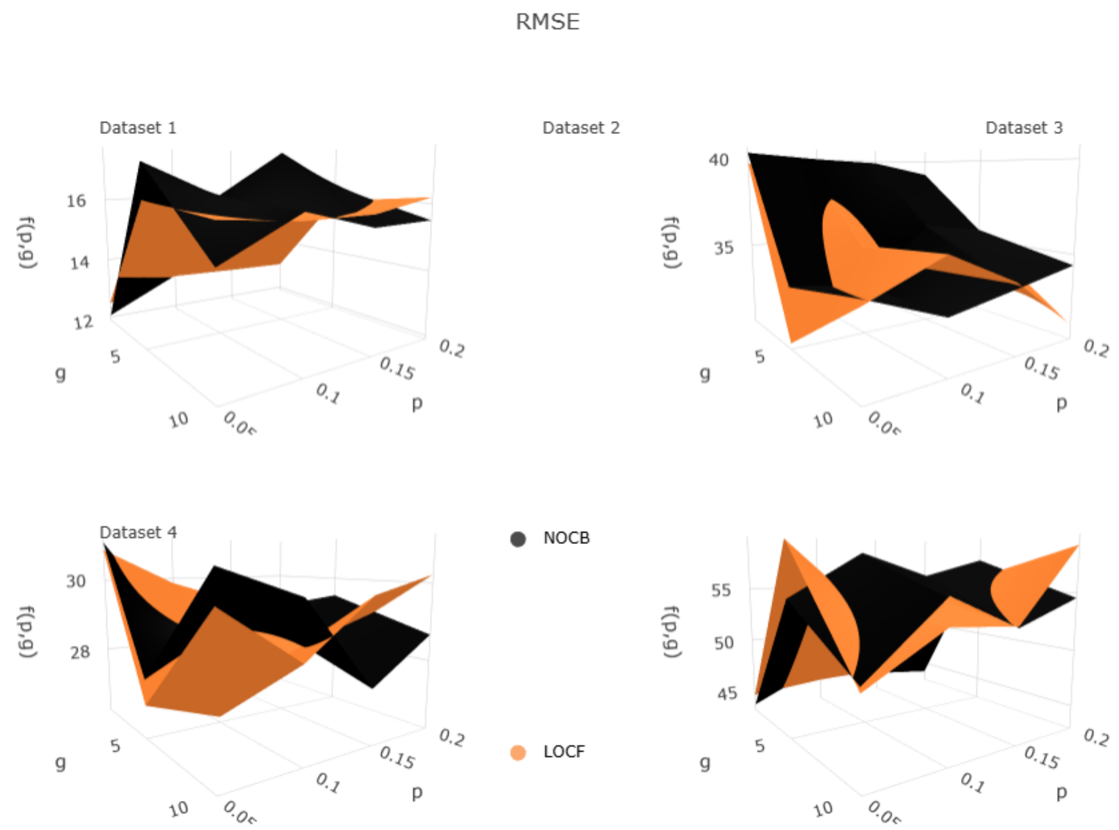


Figure 2.8: NOCB,LOCF

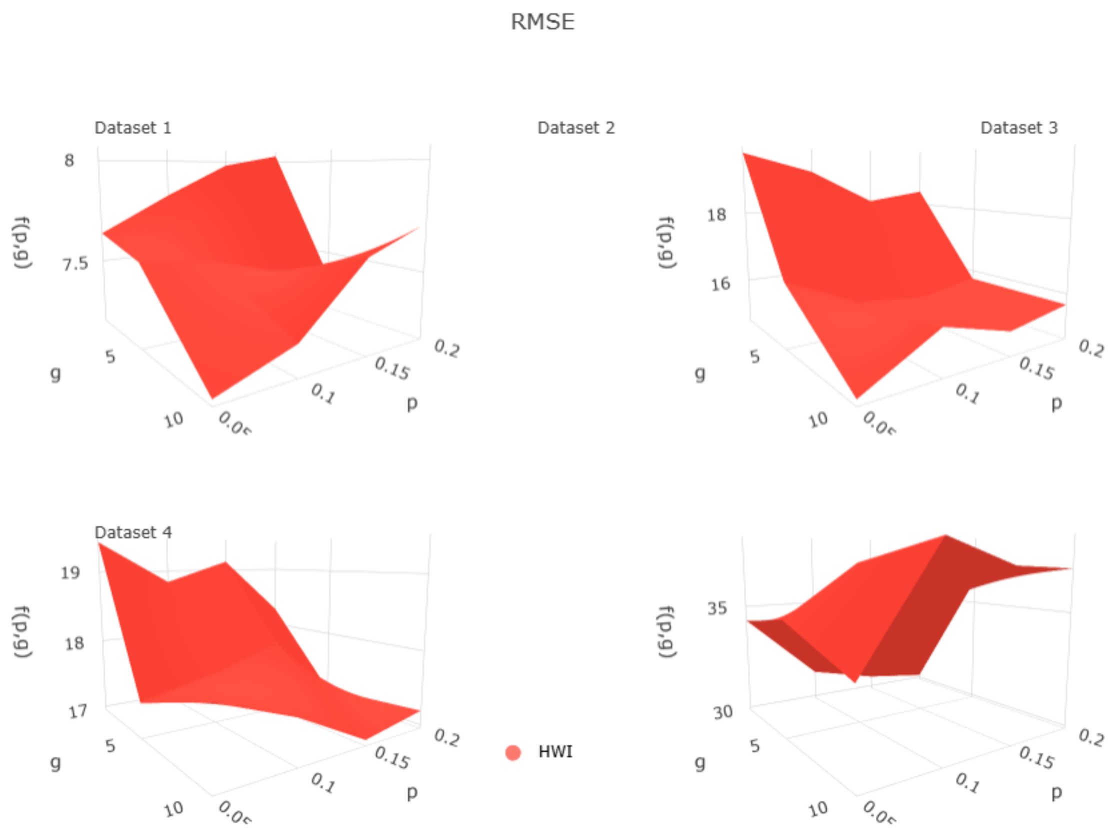


Figure 2.9: HWI

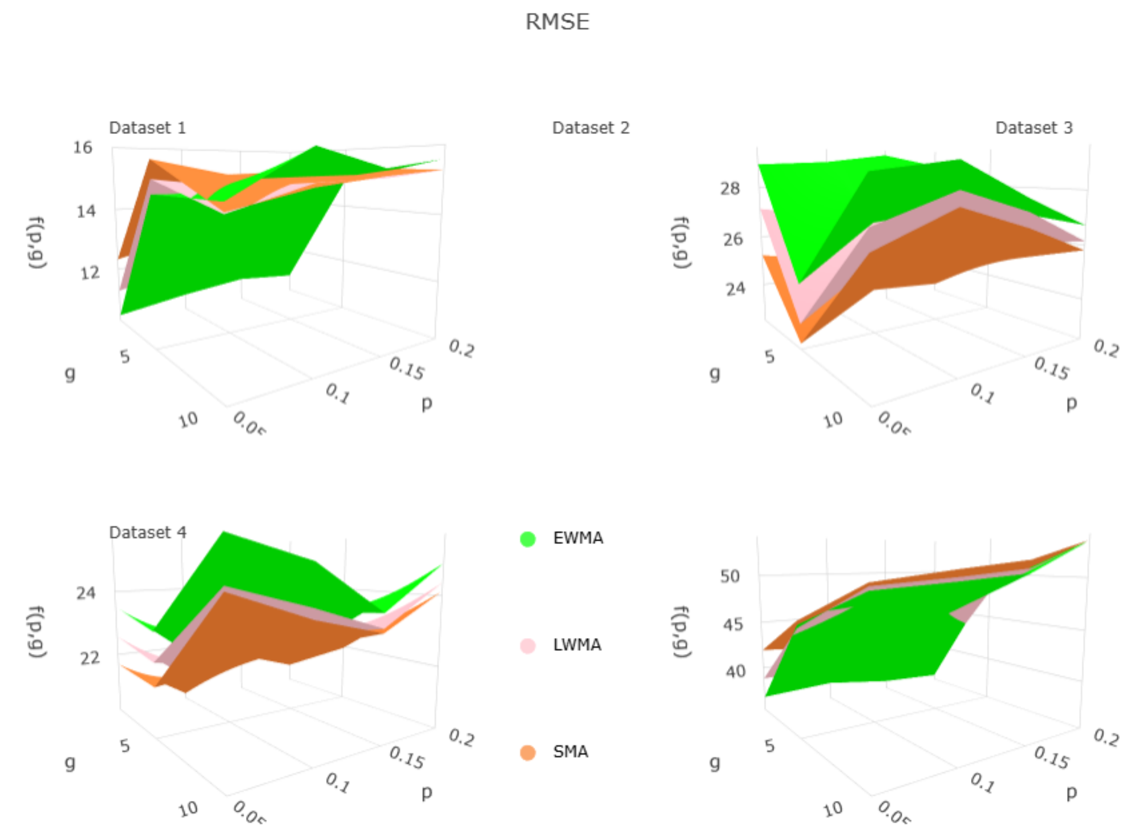


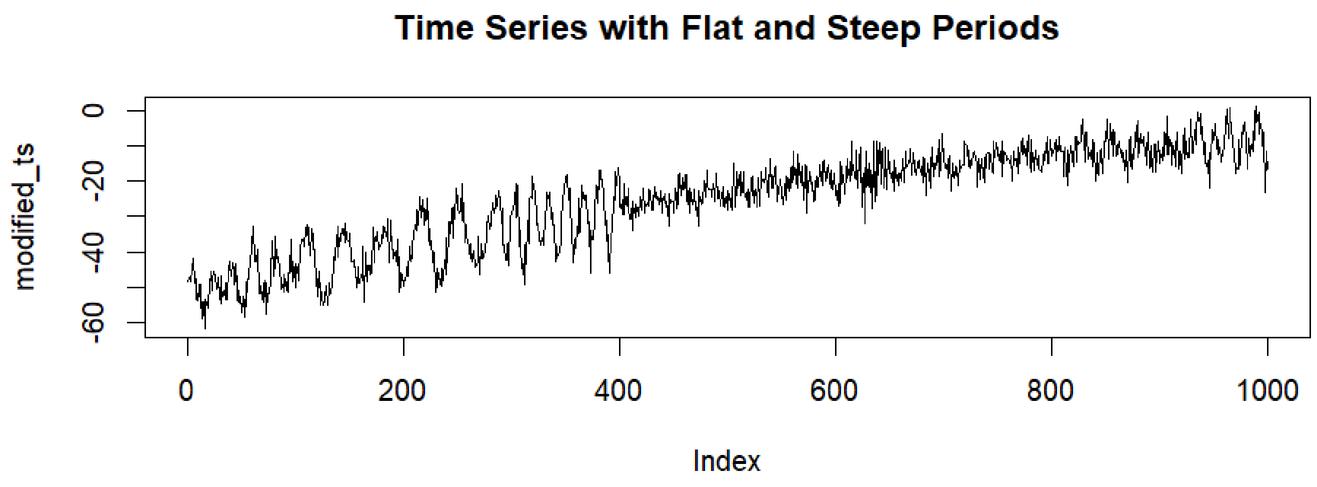
Figure 2.10: EWMA,LWMA,SMA

In this section we leave the conclusions open for now, as the fact that the frequency of the arbitrary time series we created is fixed does not fully justify the fact that the interpolation algorithm with the smallest RMSE must be the best method in any large time series. We need another example to strengthen our conjecture that the best performing HWI in this case. But before we start, I will simply talk about HWI algorithm.

Doctor Burr introduces the Hybrid Wiener Interpolator (HWI). For many classes of time series, the HWI method is optimal for interpolation performance; greatly outperforming other methods that we have discussed in this section [4]. The HWI is designed as an iterative algorithm with an Expectation-Maximization framework, building upon Brillinger's work in [2]. The method relies on a transformation of a stochastic process to and from a stationary state, using multitaper spectral estimates[8] to detect and subsequently remove trends and periodic line components from the input series. With that being said, this method assumes a time series can be modelled as a stochastic process with stationary background noise and periodic structure. The HWI plays a major role in this thesis. We directly compare the results of our newly developed method to those of the HWI with the aim of exceeding its interpolation performance.

Now let us start our second example.

Beginning with the same step as we created one-thousand-length time series, we modified this base series by extracting $X_t$, defined random segments for flat and steep periods and then shuffled them randomly to modify each segment in base time series. This step ensured that frequency and oscillation rate of modified time series changed randomly over time. The figure below showed the modified time series. This time we created 300 gappy time series for each p,g scenario, which is $K = 300$. The purpose is to interpolate the gappy time series for 300 times in order to test consistency of performance on each algorithm, for better summarizing the best algorithm. The performance is indicated as the histogram of RMSE between base and modified time series after each interpolation of algorithm.



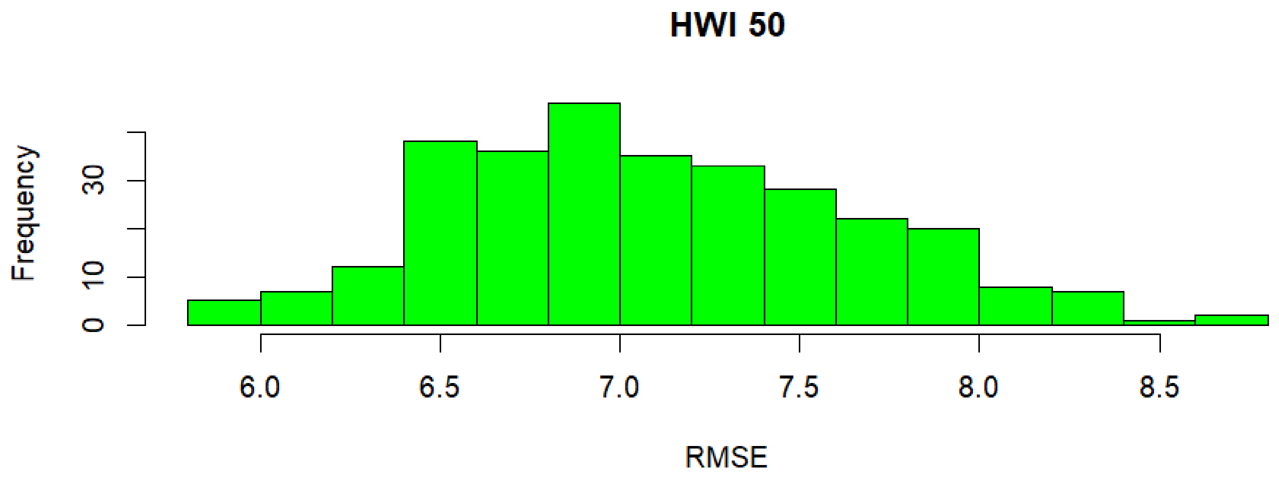
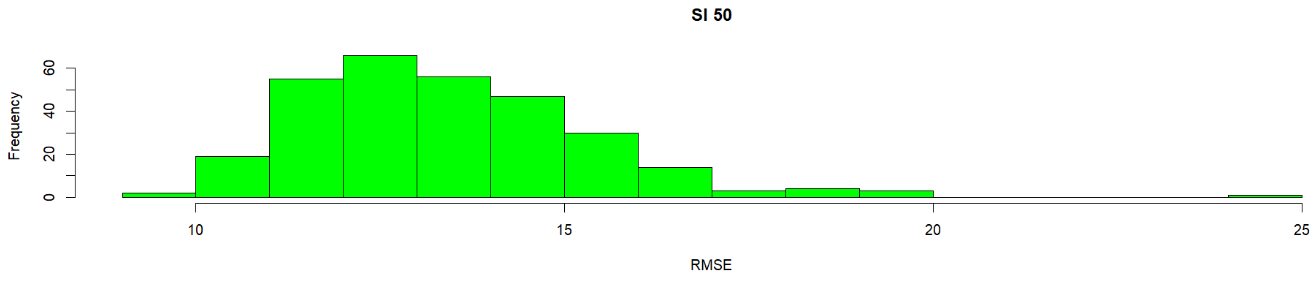


Figure 2.11: Modified Time Series and HWI



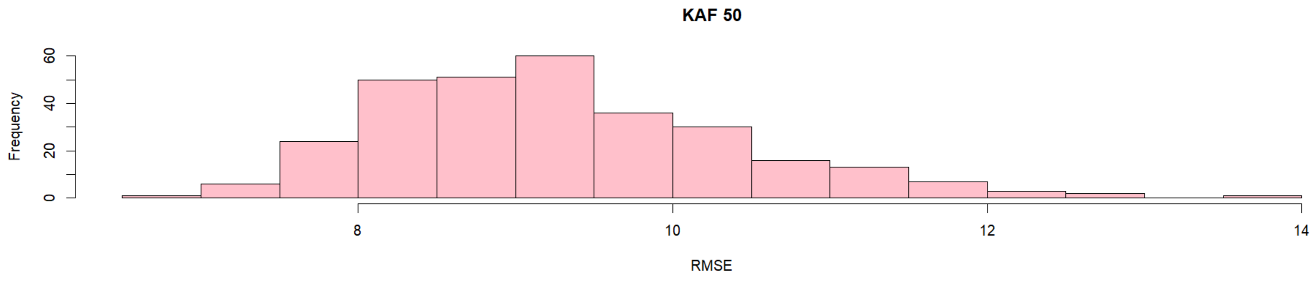


Figure 2.12: KAF,SI

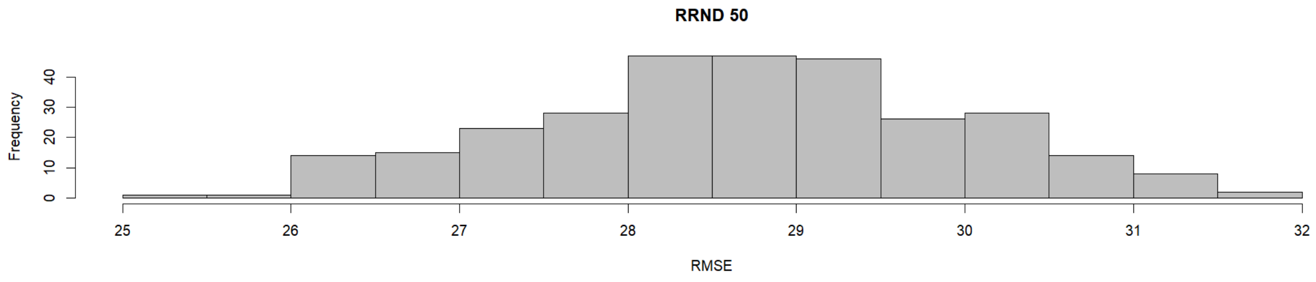
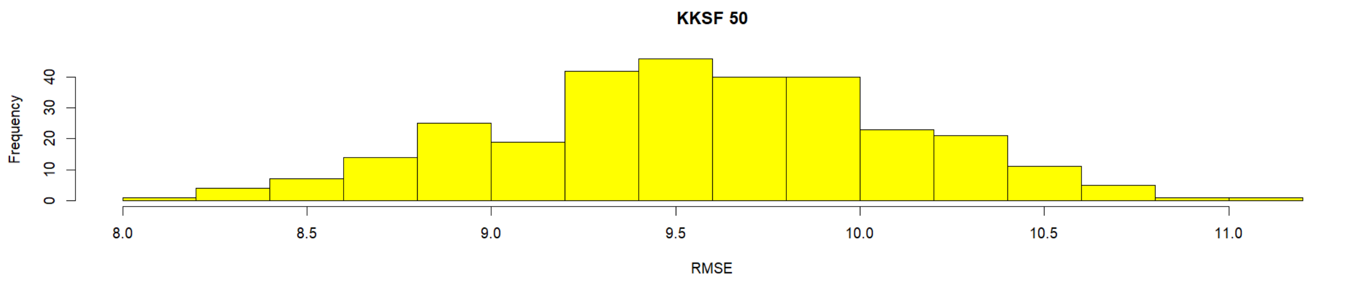


Figure 2.13: RRND



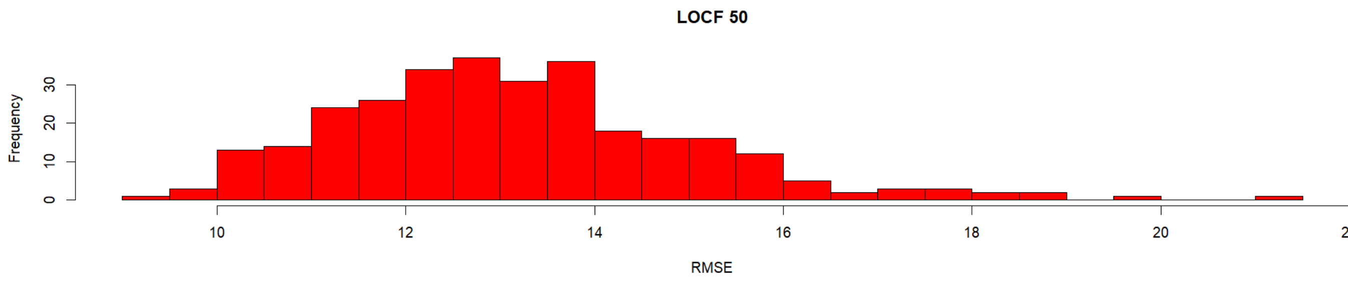
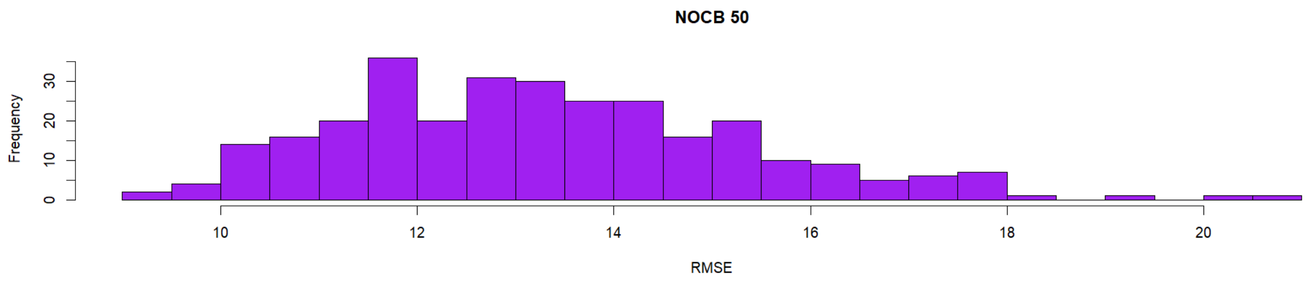


Figure 2.14: KKSF,LOCF



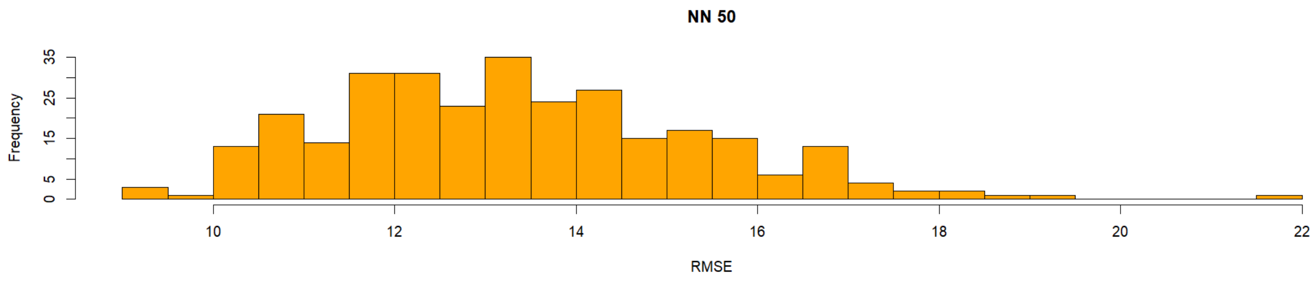


Figure 2.15: NOCB,NN

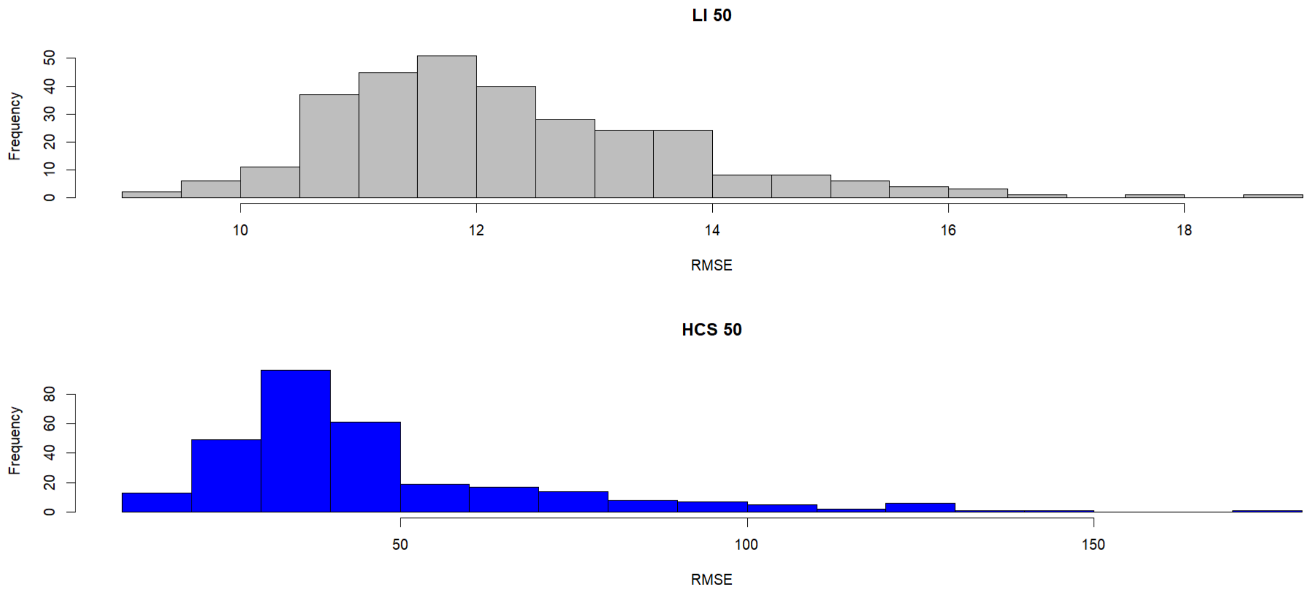


Figure 2.16: LI,HCS

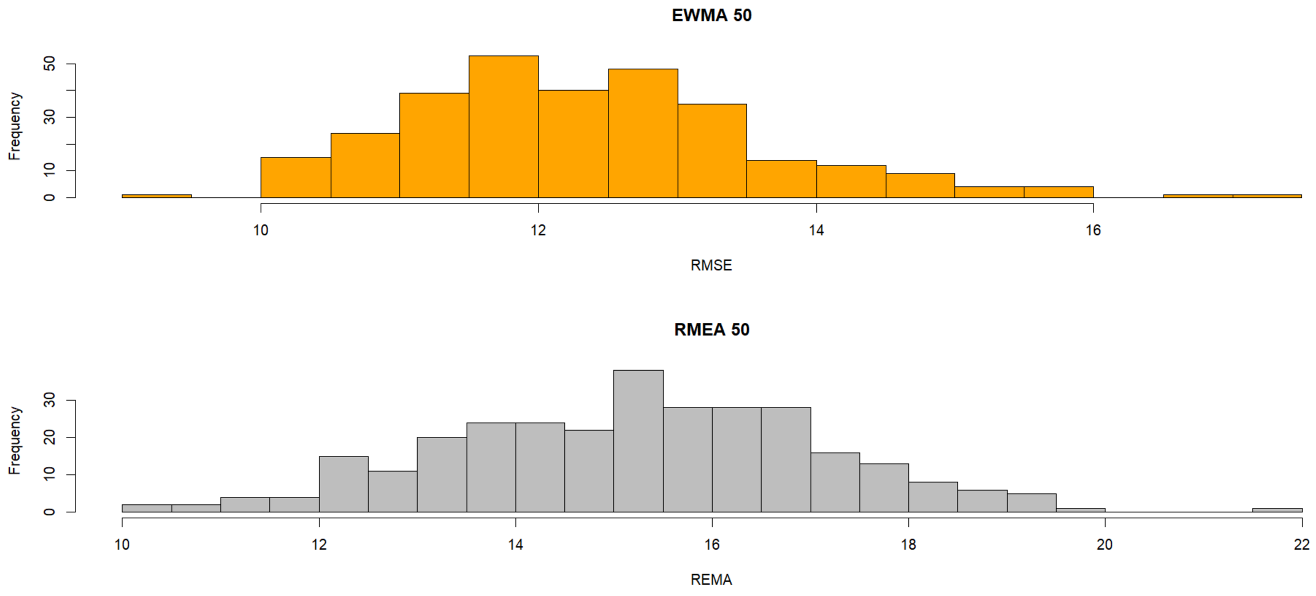


Figure 2.17: EWMA,RMEA

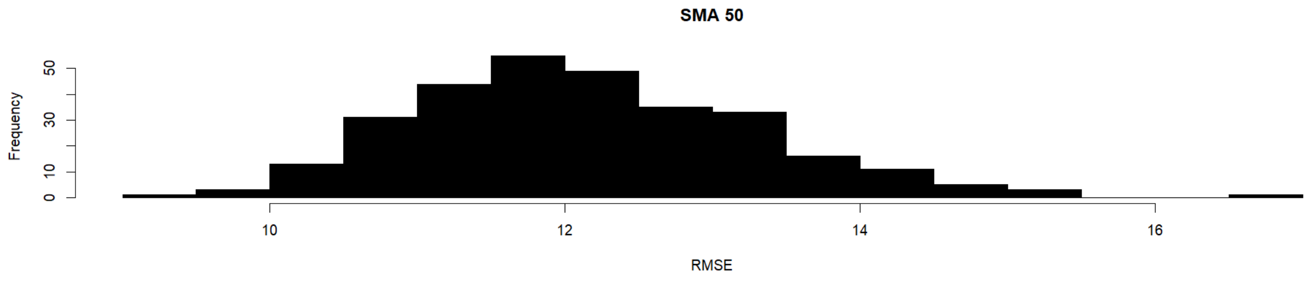


Figure 2.18: SMA,LWMA

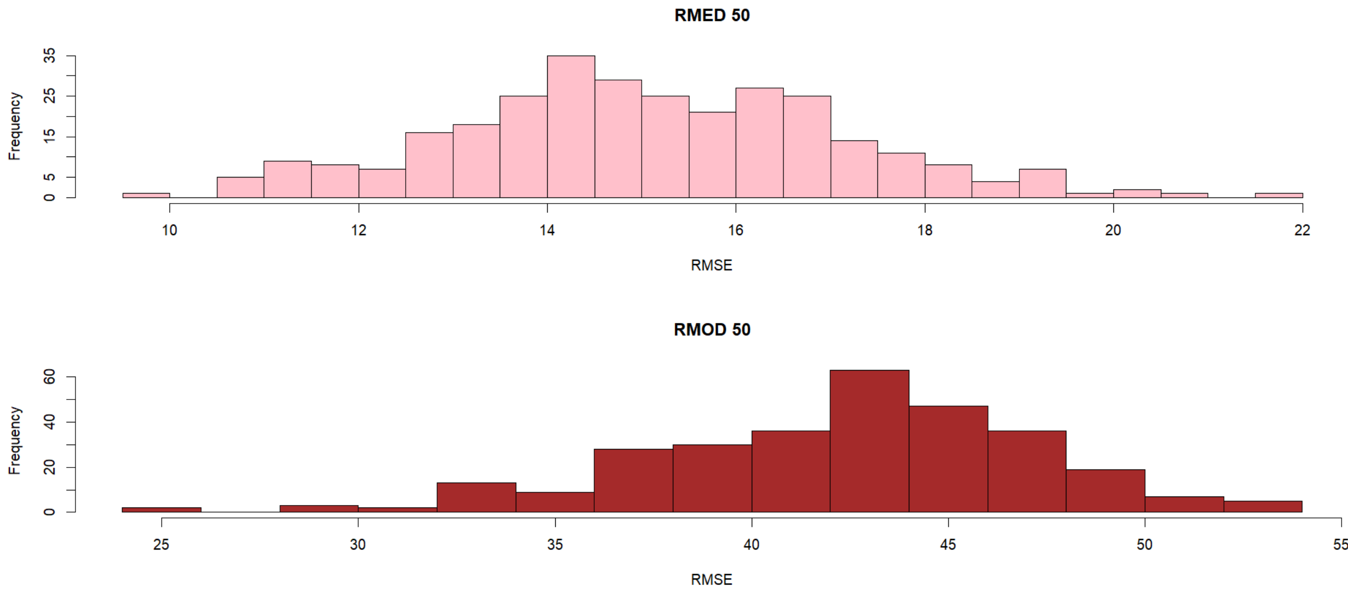


Figure 2.19: RMED,RMOD

This comprehensive approach not only illustrates the manipulation and gap-filling in time series but also evaluates the efficiency of various interpolation methods under different scenarios, providing a robust framework for statistical analysis in time series data.

## 2.3 Results of Simulation

In the first example, we could see from 3-D plots of RMSE of each algorithm, as proportion of data missing and gap length change, HWI algorithm gave the best performance. To verify the performance of HWI on more complicated scenario, we interpolated 300 time series and observed the distribution of RMSE. Like the result we saw, RMSE still performs best. KKSF and KAF algorithms are also advanced in handling non-stationary,non-linear and highly dynamic time series, making them produce robust predicition and estimation when new data arrives. In contrast, RRND and RMOD performs worst in all scenarios, which is reasonable since they are not suitable to modified time series with high oscillation rate and randomness.

# 3 Interpolation via Pattern Matching

In time series analysis, missing data poses a significant challenge for accurate modeling and forecasting. Various interpolation techniques have been developed to address this issue, with pattern matching methods emerging as a robust approach for estimating missing values. This chapter discusses the implementation and evaluation of several pattern matching interpolation techniques, each leveraging different algorithms to find the best matching segments within the time series. The goal is to assess their effectiveness in reconstructing missing data and compare their performance using Root Mean Square Error (RMSE) as a metric.

## 3.1 Data Preparation and Gap Simulation

We began with the creation of a synthetic time series to serve as the basis for testing interpolation methods. The time series is generated using a cosine function with a varying phase *phi* to introduce non-linearity. The length of time series is still 1000: $t$=1:1000, $f_0$=0.2,$phi = (t/100)^2$,and time series is $modifiedts1 = cos(2 * pi * f0 * t/10 + phi * 2 * pi)$

This base signal is further modified by adding white noise to simulate real-world variability and by introducing a polynomial trend to create a more complex, non-stationary time series: $noise$=0.3, $whitenoise$= rnorm(length(modified ts1), mean = 0, sd = noise),$ploy = (t - 500)^2/500^2 + 0.5$. The final time series is $(modifiedts1 + whitenoise) * ploy$

To simulate missing data, a gap is introduced in the time series between indices 200 and 249. This gap represents a scenario where data is unavailable, requiring interpolation to reconstruct the missing values.

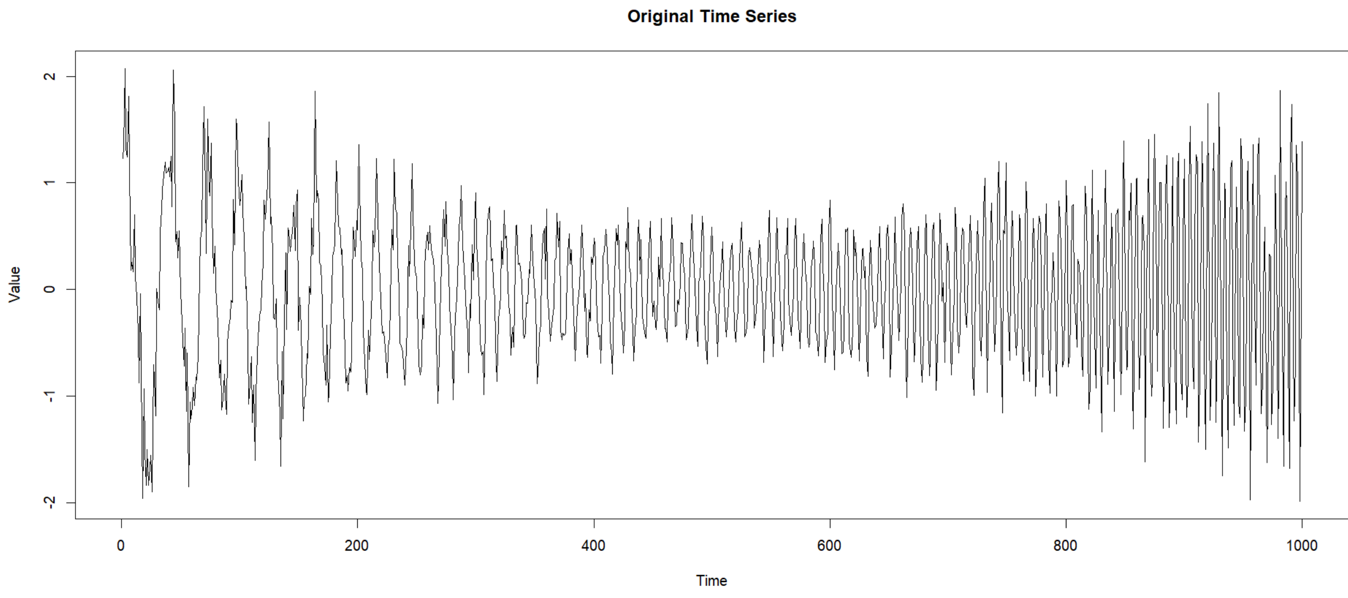


Figure 3.1: Original Time Series

## 3.2 Pattern Matching Interpolation Algorithm

Several pattern matching algorithms are implemented to fill the simulated gaps in the time series. These methods differ in how they select and weight the best matching segments from the available data.

### 3.2.1 Algorithm 1: Euclidean Distance-Based Matching

The first method involves finding the segment that most closely matches the gappy segment based on Euclidean distance. This distance is calculated between the gappy segment and all possible reference segments in the time series. The segment with the smallest Euclidean distance is selected to fill the gap.

In the beginning, to ensure sufficient context is provided for the interpolation, the algorithm defines a segment length that is four times the gap length (segment length). This longer segment length ensures that the extracted segments include enough data points on either side of the gap to facilitate accurate pattern matching.

Then, we extracted potential reference segments from the time series that do not contain any missing values (excluding the gap itself). These segments are used as candidates for pattern matching. We wrote a function iterating over the time series, extracting segments of the specified length (segment length). It then checks if these segments contain missing values outside the predefined gap indices, and only those that do not are retained for further comparison.

Next we designed another function to identify the reference segment that most closely matches the gappy segment. The function calculates the Euclidean distance between the gappy segment and each of the reference segments. The Euclidean distance is a measure of similarity, with smaller distances indicating a closer match. The reference segment with the smallest distance is selected as the best match, which will then be used to fill the gap.

In the end, we extracted the segment from the time series that includes the gap, ensuring that enough context (like surrounding data) is included on either side. The last function is then applied to identify the best matching reference segment. If a matching segment is found, the gap in the time series is filled with the corresponding values from the best matching segment. If no match is found, the algorithm halts with an error message, indicating that the interpolation could not be completed. By drawing two plots of filled time series and original time series, focusing the region around the gap, we could observe the performance of filling.

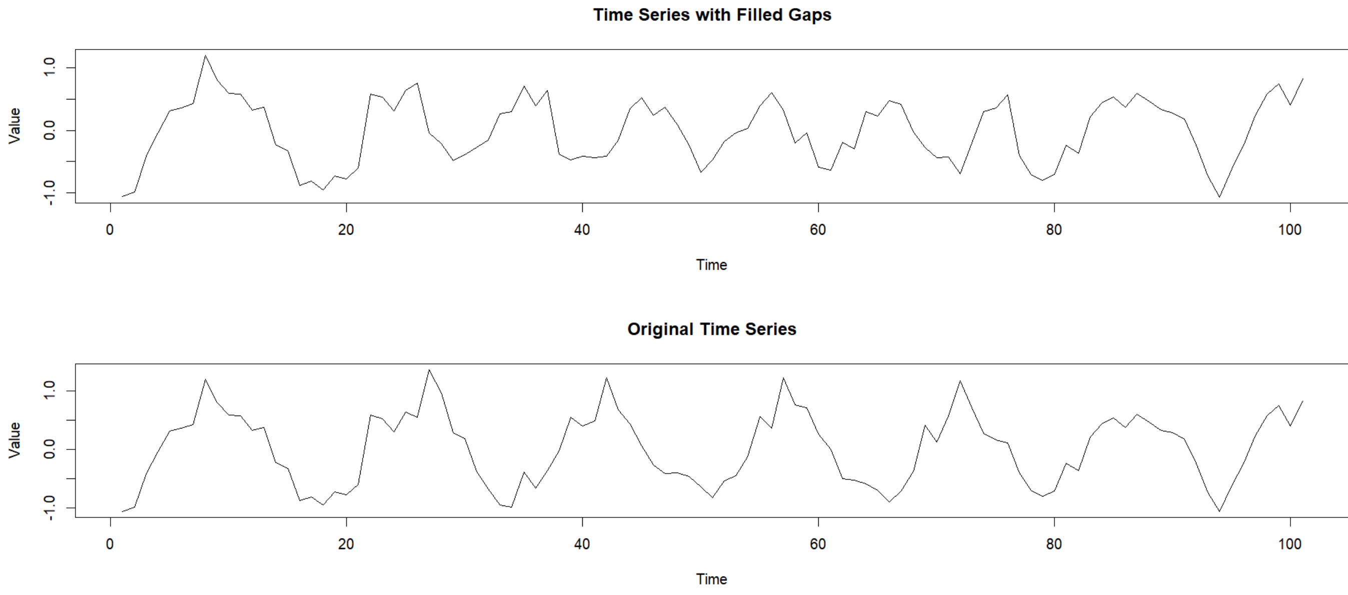


Figure 3.2: Original VS. Filled

### 3.2.2 Algorithm 2: Inverse Distance Weighting

Based on Algorithm 1, the main part of this algorithm keeps the steps of extraction of reference segment and identification of smallest segment Euclidean distance. However, we improved upon the basic Euclidean distance approach by using inverse distance weighting. Here, the contribution of each reference segment to the final imputation is inversely proportional to its distance from the gappy segment. This approach ensures that closer matches have a greater influence on the imputed values. We also tried Inverse-Square Distance Weighting, which gave even more emphasis to closely matching segments. We added a function here to perform pattern matching using an inverse distance-based weighting method. The algorithm calculated the distance between the gappy segment and each reference segment using the sum of absolute differences. These distances are then inverted (with a small value added to prevent division by zero) to determine the weights, which are normalized to sum to one. The final interpolated segment is a weighted average of all reference segments, with segments closer in distance contributing more heavily to the final estimate.

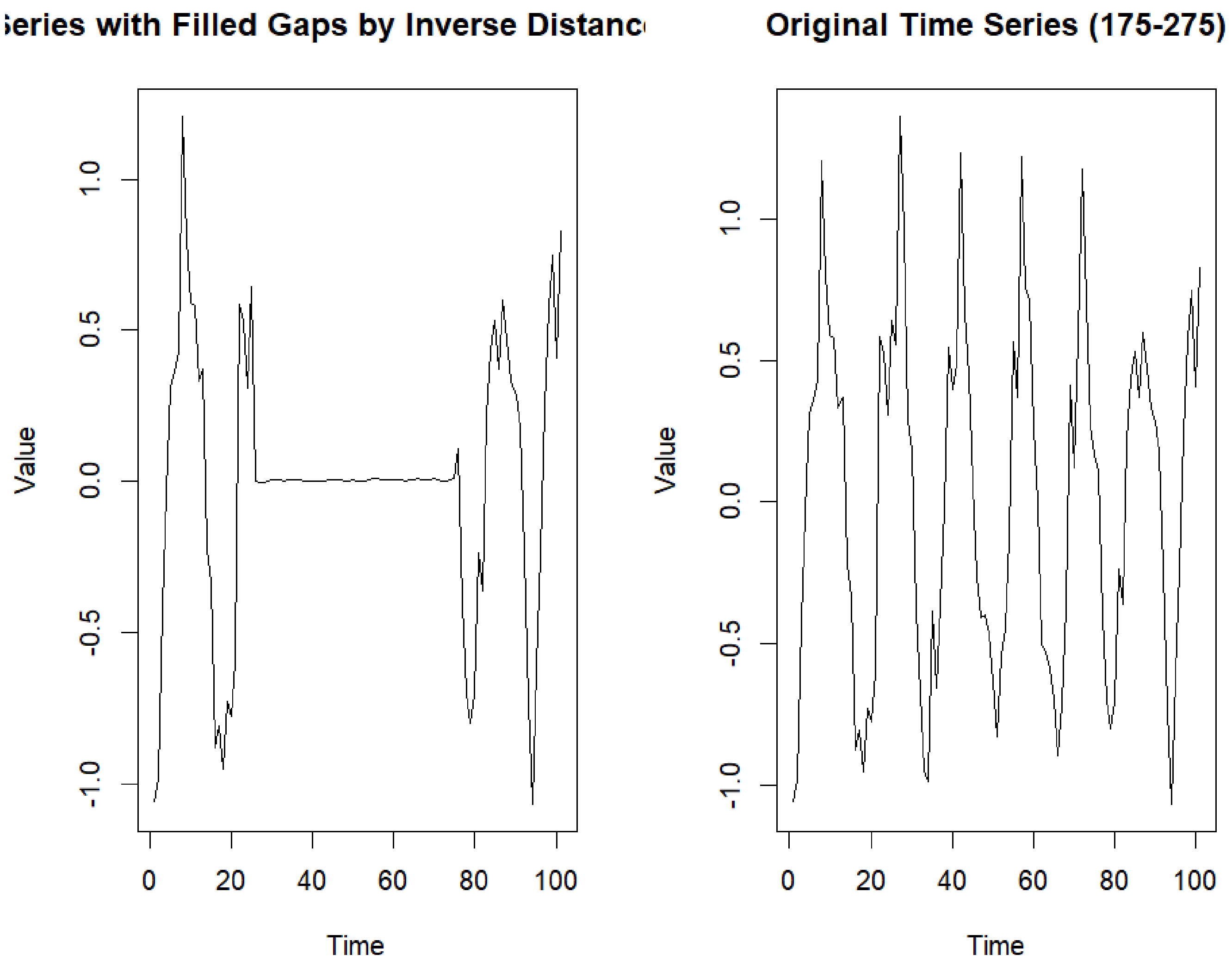


Figure 3.3: Original VS. Filled

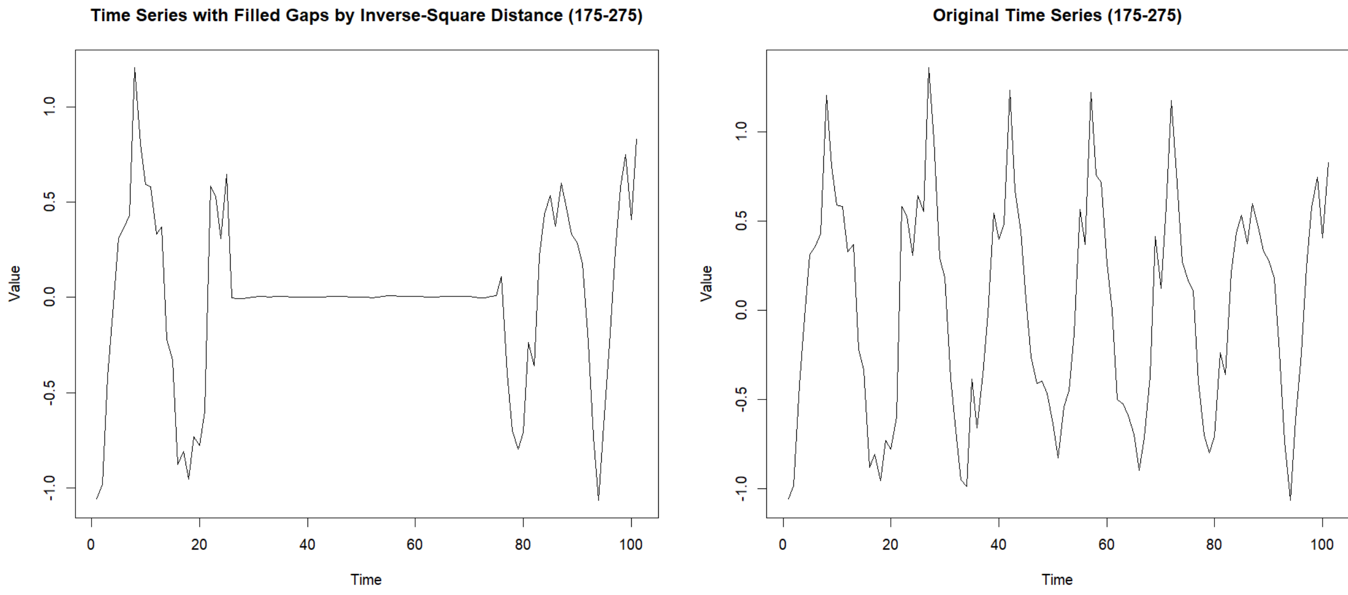


Figure 3.4: Original VS. Filled (Square)

### 3.2.3 Algorithm 3: Threshold-Based Weighting

In this method, a threshold is set based on the minimum distance found among the reference segments. Only segments within twice the minimum distance contribute to the final imputation, with equal weighting assigned to all qualifying segments. This method is tested both with and without normalization of the weights.

We added a function to implement the threshold-based pattern matching algorithm. Initially, the algorithm computes the distance between the gappy segment and each reference segment using the sum of absolute differences. The smallest distance is identified, and a threshold is set at twice this minimum distance. Only reference segments within this threshold contribute to the final interpolation, with equal weighting assigned to them. This method is tested in two variations:

Without Normalization: Where weights are not normalized, resulting in binary (0 or 1) weights.

With Normalization: Where weights are normalized to ensure they sum to one, distributing the influence of contributing segments more evenly.

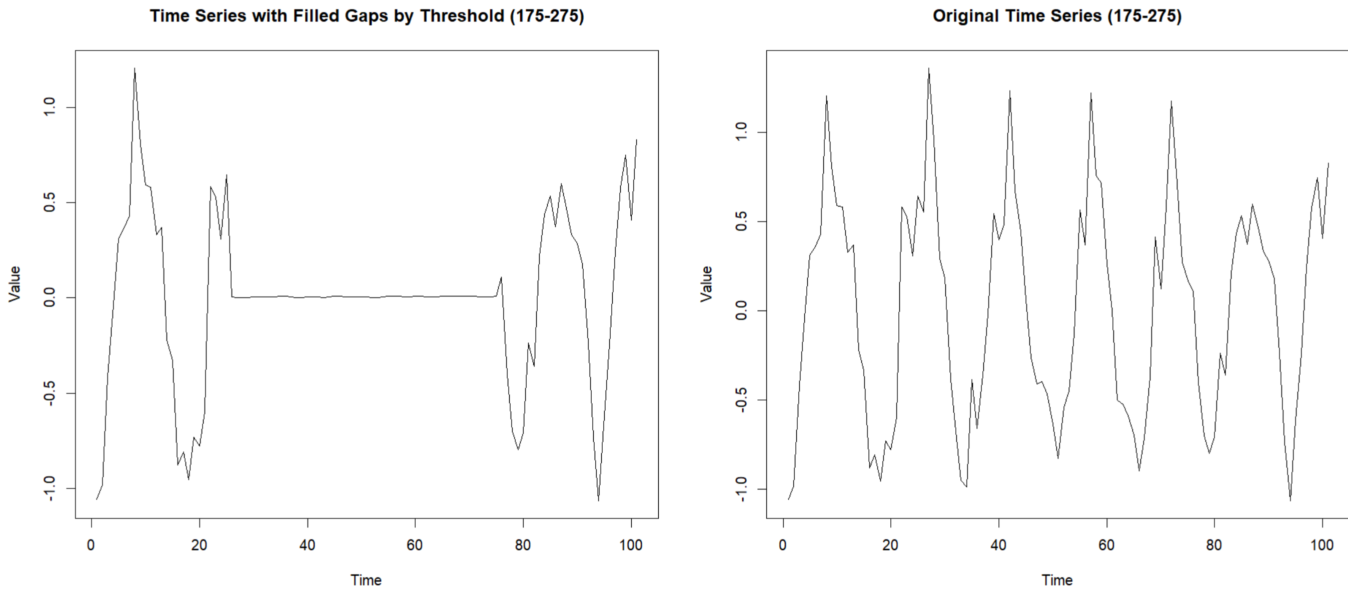


Figure 3.5: Original VS. Filled

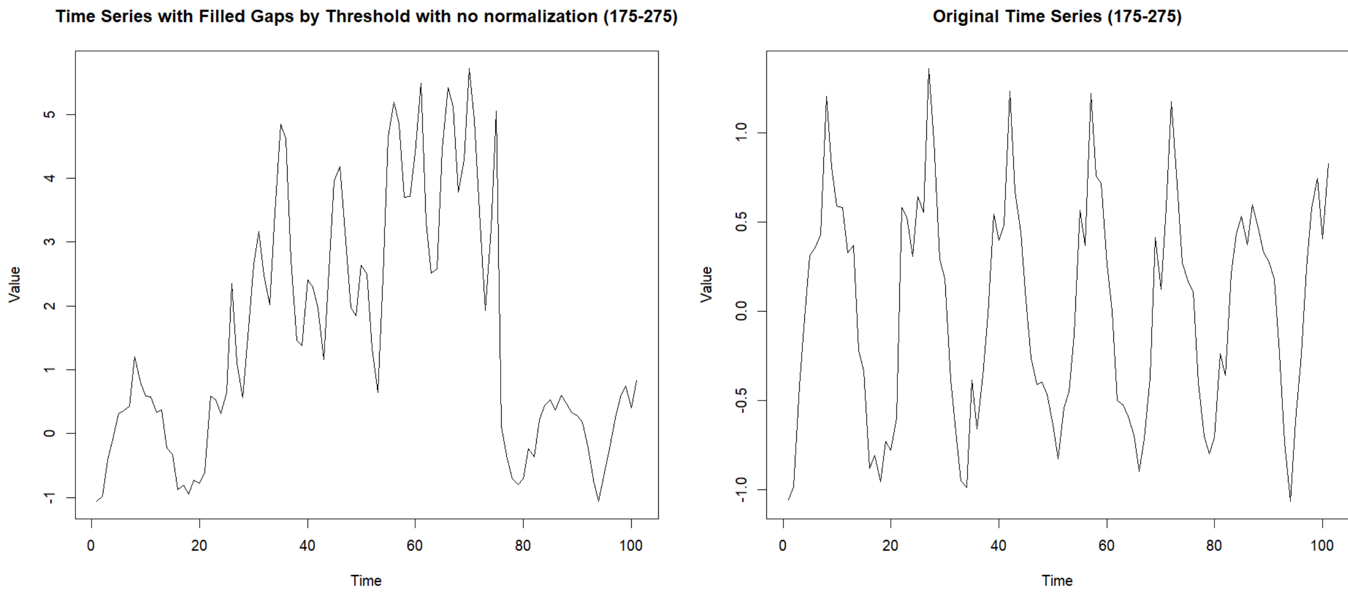


Figure 3.6: Original VS. Filled (Without Normal)

### 3.2.4 Algorithm 4: Iterative Gap Filling Using Local Segment Matching

Several of the above poor insertions have to draw our attention to the fact that picking across the entire 1000 time series data may not be rigorous, as many segments are far from the gap, and as the frequency of oscillations in the time series varies randomly, different intervals of the series possess different behaviours. However, smaller radius may fail to capture sufficient text, also leading to poor RMSE. So in the last algorithm, we filled 1 point each time and iterated 50 times, under different selection of radius.

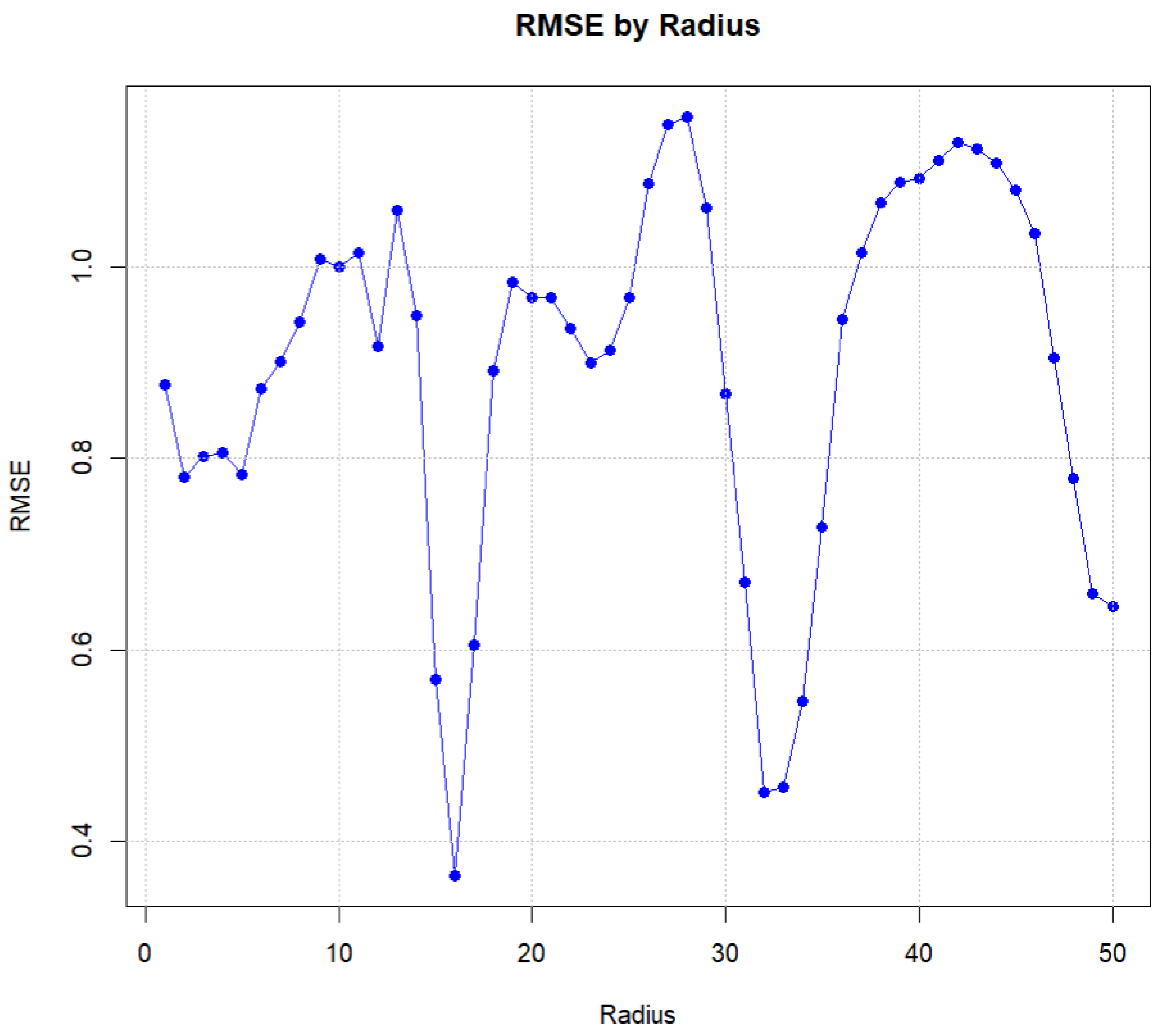


Figure 3.7: RMSE by different radii

We can see when we set the radius of reference segment equal to 16, the RMSE is smallest, compared to other radius and algorithms. When $R$=16, we plot the filling result.

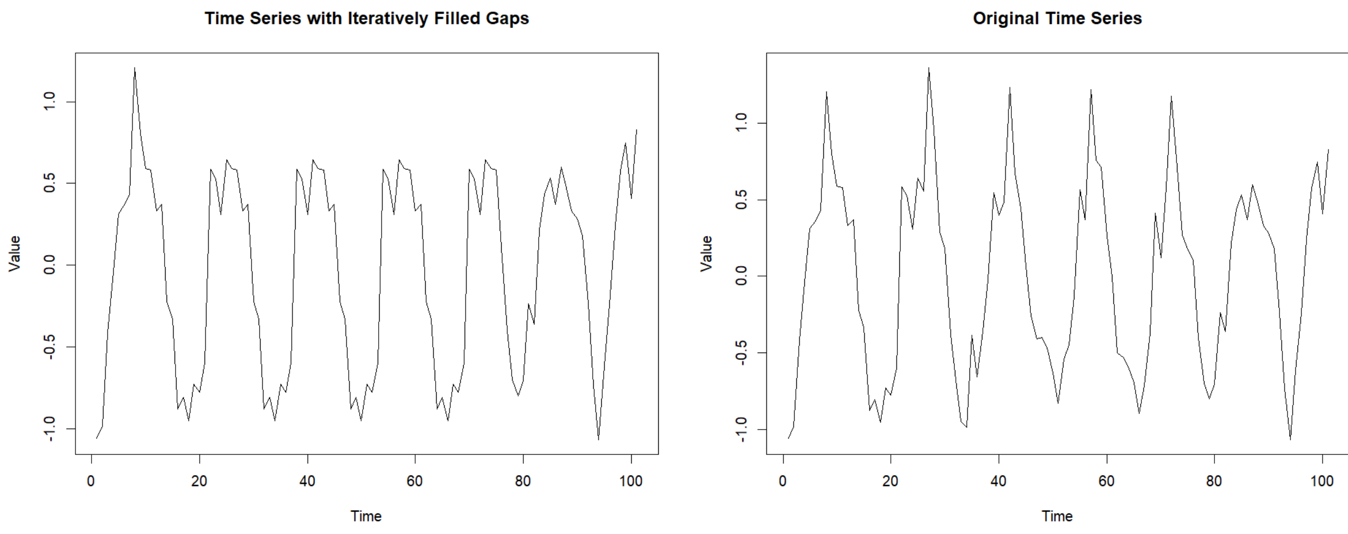


Figure 3.8: Original VS. Filled

## 3.3 Results of Simulation

| **Algorithm** | **RMSE** |
|---|---|
| Algorithm 1 | 0.816714 |
| Algorithm 2 | Inv Distance: 0.6316267 Inv-Square Distance: 0.6321384 |
| Algorithm 3 | With Normal: 0.6311504 Without Normal:3.485769 |
| Algorithm 4 | Radius = 16: 0.3634469 |

Table 3.1: Comparison of RMSE values for different algorithms

Comparing the visualization plot before and after gap fillings, we could find that Algorithm 1 to 3 did not perform well. Their RMSE between original and filling time series also prove that. Due to the too broad selection of reference segments, the steps of filling is strongly impacted by irrelevant in far interval from the gap. This is most likely why the Algorithm 2 and 3 appear to form a straight line in the middle most of the way through the filled plots, precisely because the weights are set up to exacerbate the irrelevant data and substantially diminish the impact of the relevant data. Also in Algorithm 3, without the influence of normalization, the distribution of impact of contributing segment became uneven in the filling process.

# 4 Conclusion and Future Suggestion

In the first part of research, we evaluated the performance of various interpolation algorithms across different scenarios to determine their effectiveness in filling gaps within time series data. The first set of experiments focused on a synthetic time series with a fixed frequency, where several interpolation methods were compared based on their RMSE. However, recognizing the limitations of this initial setup, we expanded our analysis to include a more complex and dynamically changing time series. This second example involved the creation of 300 gappy time series, each interpolated using different algorithms, to assess consistency and robustness in performance.

Our findings indicate that the Hybrid Wiener Interpolator (HWI) consistently outperformed other methods across both simple and complex time series scenarios. The HWI algorithm, which leverages an Expectation-Maximization framework and multitaper spectral estimates to handle non-stationary and periodic components, proved to be the most effective in minimizing RMSE, thus offering superior interpolation accuracy. This was particularly evident in more challenging scenarios where the time series exhibited high variability and oscillation rates.

In conclusion, the HWI algorithm emerges as the most reliable method for time series interpolation, especially in complex and dynamically changing datasets. The study underscores the necessity of a flexible and adaptive approach to interpolation, where the choice of algorithm is informed by the underlying structure and behavior of the data. Future work could involve further refinement of these methods and the exploration of hybrid approaches that combine the strengths of different algorithms to enhance performance in even more diverse time series contexts. I also hope that Natural Cubic Spline and Cubic Spline algorithm could be tested by following students.

In the second part, the study conducted a comprehensive evaluation of four different algorithms for interpolating gaps in a synthetic time series, with each algorithm employing distinct methods for pattern matching and data reconstruction. The goal was to identify the most effective approach for accurately filling missing data points, as measured by the Root Mean Square Error (RMSE).

In summary, the iterative approach of Algorithm 4, particularly with an optimally chosen radius, significantly outperformed the other methods in terms of both visual accuracy and RMSE. This suggests that a localized and iterative strategy is essential for accurately reconstructing missing data in time series, especially when dealing with complex and non-stationary patterns. The findings underscore the importance of considering both the selection of reference segments and the method of weighting in interpolation algorithms. Future work could focus on further refining these methods, potentially exploring dynamic adjustments to the radius or integrating additional context-aware factors to enhance interpolation accuracy across a wider range of time series data. For future students, I hope more filling and pattern matching methods could be tried since Algorithm 4 is not very accurate reflecting in the plot. Also, I think Algorithm 2 and 3 are still informative in terms of improving accuracy, depending on the choice of the reference segment.

# Bibliography

[1] [aut] Wesley S. Burr [aut] Melissa L. Van Bussel [aut, cre] Sophie Castel. castels/interptools: Tools for the systematic testing of interpolation algorithms. `https://rdrr.io/github/castels/interpTools/`, 2024.

[2] D. R. Brillinger. Statistical inference for irregularly observed processes. In David Brillinger, S. Fienberg, J. Gani, J. Hartigan, K. Krickeberg, and Emanuel Parzen, editors, *Time Series Analysis of Irregularly Observed Data*, volume 25, pages 38–57. Springer New York, New York, NY, 1984.

[3] S. Castel. A framework for testing time series interpolators. Master's thesis, Trent University (Canada), 2020.

[4] Sophie Castel and Wesley Burr. Assessing statistical performance of time series interpolators. *Engineering Proceedings*, 2021.

[5] Jim Frost. Time series analysis introduction. 2021. `https://statisticsbyjim.com/time-series/time-series-analysis-introduction/`.

[6] Kwanghoon Pio Kim Hyun Ahn, Kyunghee Sun. Comparison of missing data imputation methods in time series forecasting. *Computers, Materials Continua 2022*, (70):767–779, 2022. `https://www.techscience.com/cmc/v70n1/44403`.

[7] Kuldeep Kumar. Introduction to modern time series analysis. `https://academic.oup.com/jrsssa/article/173/1/271/7077958?login=false`, 2010.

[8] D.J. Thomson. Spectrum estimation and harmonic analysis. *Proceedings of the IEEE*, 70(9):1055–1096, 1982.