# Performance Assessment of Cubic Spline Interpolation for Weather Temperature Data Imputation

Ahmad Syafiq 13523135[1,2]
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
[1]13523135@itb.ac.id, [2]ahmad.syafiq2005@gmail.com

***Abstract*—This paper aims to evaluate the performance of cubic spline interpolation for imputing missing values in weather temperature time-series datasets. The method employs a piecewise cubic polynomial, ensuring smooth transitions between the data points while maintaining continuity and differentiability at the boundaries. The performance of the method was assessed under varying missing data patterns, rates, and temporal granularities, and compared with other commonly used imputation methods. The results demonstrate that the method can handle scattered missingness effectively, but block missingness poses greater challenges for interpolation. Scattered missingness yielded better accuracy compared to block missingness, where consecutive gaps posed greater challenges. The findings highlight the limitations of the proposed method in handling blocks and emphasize the need for more robust methods when dealing with high missingness rates or block patterns.**

***Keywords*—Cubic Spline Interpolation, Data Imputation, Missing Data, Weather Data**

## I. Introduction

Data imputation is a critical process in data analysis, especially when dealing with time-series datasets. Missing values are a common issue, often arising from sensor malfunctions, communication errors, or incomplete data collection processes. If not addressed, these gaps can disrupt the temporal continuity of the dataset, leading to biased results, reduced statistical power, and invalid conclusions [1]. Effective imputation is essential to preserve the representativeness of the dataset, enabling accurate analyses by replacing missing values with reasonable estimates that maintain the integrity of the time-series structure.

A wide range of imputation techniques has been developed to address missing data in time-series and other types of datasets. Simple methods, such as mean or median imputation, are widely used due to their ease of implementation. However, these methods can fail to account for temporal patterns and may introduce bias or reduce variability, particularly when data are not missing completely at random. More advanced approaches, such as regression imputation, utilize the relationships between variables to produce estimates that better reflect the dataset's underlying structure and variability. Additionally, machine learning-based methods like k-Nearest Neighbors (k-NN), decision trees, and neural networks have gained popularity for their ability to model complex patterns in time-series data. Despite their accuracy, these methods often require substantial computational resources and may be impractical for large-scale or resource-constrained applications.

Cubic spline interpolation, a technique that constructs piecewise cubic polynomials between data points, is particularly well-suited for imputing continuous variables in time-series data. By creating a smooth curve that passes through the known data points, this method strikes a balance between simplicity and accuracy, producing estimates that align closely with the underlying trends. The smoothness of cubic spline interpolation makes it especially effective in scenarios where temporal continuity is a defining characteristic of the dataset.. However, the performance of cubic spline interpolation in the context of temperature data imputation remains underexplored.

Weather temperature datasets, as a type of time-series data, present both opportunities and challenges for imputation methods. Their smooth temporal patterns, characterized by gradual changes over time, make them an ideal candidate for spline-based interpolation, which effectively captures these patterns by fitting smooth curves through known data points. Spline interpolation leverages the inherent continuity in temperature changes, providing accurate estimates for missing values, especially in scenarios where data gaps are relatively small or evenly distributed.

This paper aims to evaluate the performance of cubic spline interpolation for imputing missing values in weather temperature time-series datasets. By examining its effectiveness under varying missing data patterns, rates, and temporal granularities this research provides insights into the method's applicability and limitations. These findings aim to inform the use of cubic spline

interpolation in time-series data imputation and other similar contexts.

## II. Cubic Spline Interpolation

Cubic spline interpolation is a method for approximating a smooth curve that passes through a set of known points. It divides the curve into segments and fits a cubic polynomial in each interval between two consecutive points. The method ensures the curve is smooth and continuous up to the second derivative.

For $n+1$ points $(x_0, y_0)$, $(x_1, y_1)$,..., $(x_n, y_n)$, the cubic polynomial for the interval $[x_i, x_{i+1}]$ is expressed as:

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

,

where $a_i, b_i, c_i, d_i$ are coefficients to be determined.

The constraints on the spline are:

1. The spline passes through all data points:
   $S_i(x_i) = y_i$ and $S_i(x_{i+1}) = y_{i+1}$.
2. The first and second derivatives are continuous:
   $S'_i(x_{i+1}) = S'_{i+1}(x_{i+1})$,
   $S''_i(x_{i+1}) = S''_{i+1}(x_{i+1})$.
3. Boundary conditions are applied to define the behavior at $x_0$ and $x_n$:
   - Natural spline: $S''_0(x_0) = 0$ and $S''_{n-1}(x_n) = 0$
   - Clamped spline: Specify $S'_0(x_0)$ and $S'_{n-1}(x_n)$.

From the constraints, the second derivatives $c_i$ at each point are found by solving a tridiagonal linear system. For $n-1$ interior points, the system is given by:

$$\frac{h_{i-1}}{6}c_{i-1} + \frac{h_{i-1}+h_i}{3}c_i + \frac{h_i}{6}c_{i+1} = \frac{y_{i+1}-y_i}{h_i} - \frac{y_i-y_{i-1}}{h_{i-1}},$$

where $h_i = x_{i+1} - x_i$.

Then, the equations can be written in matrix form:

$$\mathbf{Ac = b},$$

where **A** is a tridiagonal matrix, $\mathbf{c} = [c_1, c_2, ..., c_{n-1}]^T$ are the unknown second derivatives, and **b** is the right-hand side vector derived from the given data

After solving $\mathbf{Ac} = \mathbf{b}$, the coefficients of each polynomial are computed:

$$b_i = \frac{y_{i+1}-y_i}{h_i} - \frac{h_i}{3}(2c_i + c_{i+1}),$$

$$d_i = \frac{c_{i+1}-c_i}{3h_i},\ a_i = y_i.$$

The result of the cubic spline is constructed as a piecewise function:

$$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1], \\ S_1(x), & x \in [x_1, x_2], \\ \vdots & \\ S_{n-1}(x), & x \in [x_{n-1}, x_n]. \end{cases}$$

Fig 2.1 Cubic spline interpolation represented as a piecewise function over defined subintervals.

## III. Data Imputation

Data imputation is a statistical technique used to estimate and fill in missing values in datasets. Missing data can arise from various factors, such as equipment failures, human error, or external disruptions. The presence of missing data can compromise the integrity of analyses, leading to biased results or reduced statistical power if not addressed appropriately. Imputation seeks to mitigate these issues by providing plausible values for the missing entries, enabling complete and consistent datasets for analysis.

### *A. Types of Missing Data*

Understanding the underlying mechanism of missing data is crucial for choosing the appropriate imputation method. The three primary types of missing data are:

1. Missing Completely At Random (MCAR):
   The probability of missingness is independent of observed or unobserved data. For example, sensor failure unrelated to temperature values.
2. Missing At Random (MAR):
   The probability of missingness is related to observed data but not to the unobserved missing values themselves. For instance, temperature readings might be missing more frequently during specific hours of the day.
3. Missing Not At Random (MNAR):
   The probability of missingness depends on the unobserved data. For example, extreme temperature values might be systematically missed due to sensor limitations.

This paper focuses on MCAR mechanisms, as it allows for unbiased imputation without requiring strong assumptions about the missingness process.

### *B. Imputation Methods*

Various techniques have been developed for imputing missing data, ranging from simple to advanced methods:

1. Mean/Median Imputation:
   Missing values are replaced by the mean or median of the observed data. While simple, this method can distort the distribution of the data and reduce variability.
2. Linear Interpolation:
   This method estimates missing values by fitting a

straight line between known data points. It works well for continuous data with relatively short gaps.

3. Multiple Imputation:
   A statistical approach that generates multiple plausible datasets by imputing missing values using predictive models, then combines the results for robust analysis.
4. Machine Learning-Based Imputation:
   Techniques such as k-nearest neighbors, random forests, and deep learning can predict missing values based on patterns in the data. These methods are particularly useful for complex datasets with nonlinear relationships.

## V. Methodology

This paper evaluates the performance of cubic spline interpolation for imputing missing values in weather temperature datasets. The methodology is structured into four main stages: data collection, preprocessing, interpolation, and evaluation.

### *A. Data Collection*

Weather temperature data were sourced using the Meteostat library, which provides access to historical meteorogical data. The data for this paper were obtained for a specific location corresponding to Bandung, Indonesia. To assess the interpolation method's performance under different temporal granularities, both daily and hourly temperature data were collected. The time periods varied, including short-term (7 days), medium-term (3 months), and long-term (1 year), which allowed for a comprehensive evaluation of cubic spline interpolation under different missing data densities and seasonal patterns.

The following Python code snippet illustrates the data collection process:

```python
from datetime import datetime
from meteostat import Point, Daily, Hourly

location = Point(-6.9389, 107.7528)
start = datetime(2023, 1, 1, 0, 0, 0)
short_term_end = datetime(2023, 1, 7, 23, 59, 59) # 7 days
medium_term_end = datetime(2023, 3, 31, 23, 59, 59) # 3 months
long_term_end = datetime(2023, 12, 31, 23, 59, 59) # 1 year

daily_long_data = Daily(location, start, long_term_end)
daily_long_data = daily_long_data.fetch()

hourly_long_data = Hourly(location, start, long_term_end)
hourly_long_data = hourly_long_data.fetch()

daily_long_temp = daily_long_data['tavg']
daily_medium_temp = daily_long_temp[start:medium_term_end]
hourly_long_temp = hourly_long_data['temp']
hourly_medium_temp = hourly_long_temp[start:medium_term_end]
hourly_short_temp = hourly_long_temp[start:short_term_end]
print(daily_long_temp)
print(hourly_long_temp)
print("\nMissing data:")
print("Daily:", daily_long_temp.isnull().sum())
print("Hourly:", hourly_long_temp.isnull().sum())
```

✓ 3.2s

Fig 5.1 Python code for temperature data collection using the Meteostat library.

### *B. Data Preprocessing*

To evaluate the performance of cubic spline interpolation, missing data were artificially introduced into both the hourly and daily datasets. This step aimed to simulate common real-world data loss scenarios, such as sensor failure or incomplete reporting. The creation of artificial missing data was designed to represent varying levels and patterns of missingness. Two distinct missing data patterns were applied: scattered missingness and block missingness. Both missing data patterns are based on the Missing Completely At Random (MCAR) assumption. Missing data were introduced at three different rates: 10%, 20%, and 30% of the total data points. The following preprocessing steps were implemented:

1. Scattered Missingness:
   Missing values were randomly distributed across the dataset, independent of the data values themselves.
2. Block Missingness:
   Missing values were introduced in contiguous blocks (sequential periods) within the dataset to simulate continuous gaps. Blocks of missing values were randomly placed within the time series. The length of each block was set to 12 hours for hourly data and 7 days for daily data.
3. Edge Exclusion:
   Datasets with missing values at the edges were excluded, as interpolation methods require valid data points on both sides of the missing values. This ensures that the interpolation is evaluated on properly placed missing values within the dataset.

The following Python code snippet illustrates the missing data introduction process:

```python
def introduce_missingness(data, rate, pattern="scattered", block_size=12):
    data_with_nan = data.copy()
    n = len(data)
    missing_indices = []
    if pattern == "scattered":
        missing_indices = np.random.choice(n, int(rate * n), replace=False)
    elif pattern == "block":
        num_blocks = int(rate * n / block_size)
        for _ in range(num_blocks):
            start_idx = np.random.randint(0, n - block_size)
            missing_indices.extend(range(start_idx, start_idx + block_size))
    data_with_nan.iloc[missing_indices] = np.nan
    return data_with_nan

missing_rates = [0.1, 0.2, 0.3]
patterns = ["scattered", "block"]

def create_missing_datasets(data, patterns, rates, block_size):
    datasets = {}
    for pattern in patterns:
        for rate in rates:
            key = f"{pattern}_{int(rate*100)}"
            datasets[key] = introduce_missingness(data, rate, pattern, block_size)
    return datasets

hourly_missing_data_long = create_missing_datasets(hourly_long_temp, patterns, missing_rates, block_size=12)
daily_missing_data_long = create_missing_datasets(daily_long_temp, patterns, missing_rates, block_size=7)
hourly_missing_data_medium = create_missing_datasets(hourly_medium_temp, patterns, missing_rates, block_size=12)
daily_missing_data_medium = create_missing_datasets(daily_medium_temp, patterns, missing_rates, block_size=7)
hourly_missing_data_short = create_missing_datasets(hourly_short_temp, patterns, missing_rates, block_size=12)
```

✓ 0.0s

Fig 5.2 Python code for missing data (scattered and block missingness) introduction.

### *C. Interpolation*

Cubic spline interpolation was applied to the datasets with missing values to estimate the missing entries and restore the continuity of the time-series data. This method constructs a piecewise cubic polynomial, ensuring smooth transitions between the data points while maintaining continuity and differentiability at the boundaries. The piecewise nature of the cubic spline allows it to adapt to the local structure of the data, producing accurate and reliable imputed values that align with the overall trends.

The scipy.interpolate library in Python was employed to perform the cubic spline interpolation. Specifically, the CubicSpline class from this library was used to generate a smooth curve through the observed data points. Missing values were imputed based on the surrounding observed data, leveraging the temporal continuity inherent in the weather temperature datasets. This interpolation was conducted separately for both the hourly and daily datasets, taking into account the specific temporal resolution of each dataset to ensure the imputed values remained consistent with the underlying patterns.

The following Python code snippet provides an example of how cubic spline interpolation was performed:

```python
def cubic_spline_interpolation(original_data, missing_data):
    interpolated_data = missing_data.copy()
    observed_indices = ~missing_data.isna()
    observed_x = np.arange(len(original_data))[observed_indices]
    observed_y = original_data[observed_indices]

    spline = CubicSpline(observed_x, observed_y)
    interpolated_data[~observed_indices] = spline(np.arange(len(original_data))[~observed_indices])

    return interpolated_data
✓ 0.0s
```

Fig 5.3 Python code for cubic spline interpolation implementation.

## *D. Performance Evaluation*

To evaluate the accuracy of the imputation, the performance of cubic spline interpolation was measured using two commonly used metrics: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). These metrics were calculated by comparing the imputed values with the original data. The formulas for RMSE are as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \widehat{x_i})^2}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|x_i - \widehat{x_i}|$$

Here, $x_i$ represents the true value, $\widehat{x_i}$ represents the imputed value, and $n$ is the number of data points.

The following Python code snippet illustrates the performance evaluation process:

```python
def evaluate_interpolation(original_data, interpolated_data, missing_data):
    true_values = original_data[missing_data.isna()]
    imputed_values = interpolated_data[missing_data.isna()]

    rmse = np.sqrt(mean_squared_error(true_values, imputed_values))
    mae = mean_absolute_error(true_values, imputed_values)
    return rmse, mae
results = []
for dataset_type, datasets, original_data in zip(
    ["Hourly", "Daily", "Hourly", "Daily", "Hourly"],
    [
        hourly_missing_data_long, daily_missing_data_long,
        hourly_missing_data_medium, daily_missing_data_medium,
        hourly_missing_data_short
    ],
    [
        hourly_long_temp, daily_long_temp,
        hourly_medium_temp, daily_medium_temp,
        hourly_short_temp
    ]
):
    granularity = (
        "long" if id(original_data) in [id(hourly_long_temp), id(daily_long_temp)] else
        "medium" if id(original_data) in [id(hourly_medium_temp), id(daily_medium_temp)] else
        "short"
    )
    for key, missing_data in datasets.items():
        interpolated_data = cubic_spline_interpolation(original_data, missing_data)
        rmse, mae = evaluate_interpolation(original_data, interpolated_data, missing_data)
        results.append([dataset_type, key.split("_")[0], key.split("_")[1], granularity, rmse, mae])

results_df = pd.DataFrame(results, columns=["Type", "Pattern", "Rate (%)", "Granularity", "RMSE", "MAE"])
print(results_df)
✓ 0.2s
```

Fig 5.4 Python code for interpolation result evaluation.

# VI. Results

This section presents the performance evaluation of the cubic spline interpolation method across various missingness scenarios. The results are summarized in Tables 1–4, which report the Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) under different missing data patterns and data granularities. The findings provide insights into the impact of missingness patterns, rates, and temporal granularities on interpolation accuracy.

TABLE I. Interpolation Errors for Hourly Data with Scattered Missingness

| **Granularity** | **RMSE (℃)** | | | **MAE (℃)** | | |
|---|---|---|---|---|---|---|
| | ***10%*** | ***20%*** | ***30%*** | ***10%*** | ***20%*** | ***30%*** |
| 1 year | 1.927 739 | 2.158 186 | 2.606 162 | 1.446 166 | 1.623 185 | 1.803 609 |
| 3 months | 2.119 266 | 2.255 280 | 2.479 978 | 1.656 264 | 1.723 923 | 1.859 630 |
| 7 days | 1.320 718 | 2.101 309 | 2.777 084 | 1.111 440 | 1.617 144 | 2.062 001 |

Table I illustrates the interpolation errors for hourly data with scattered missingness across three rates: 10%, 20%, and 30%. The RMSE and MAE values increase consistently with the missingness rate, demonstrating the sensitivity of the method to the proportion of missing data. This trend is observed across all temporal granularities (1 year, 3 months, and 7 days).

For instance, for a 1-year granularity at a 10% missingness rate, the RMSE is 1.9277 ℃, which rises to 2.6062 ℃ at a 30% missingness rate. Similarly, the MAE increases from 1.4462 ℃ to 1.8036 ℃. Shorter time spans, such as 7 days, exhibit lower errors at lower missingness rates, with an RMSE of 1.3207 ℃ for 10% missingness. However, as the missingness rate increases to 30%, the RMSE rises to 2.7771 ℃, highlighting the compounding effect of increased missingness over shorter granularities.

These results indicate that scattered missingness in hourly data poses challenges for cubic spline interpolation, particularly as the missingness rate increases. However, shorter temporal granularities offer relatively lower errors, suggesting the method's effectiveness when data is densely sampled.

TABLE II. Interpolation Errors for Hourly Data with Block Missingness

| **Granularity** | **RMSE (℃)** | | | **MAE (℃)** | | |
|---|---|---|---|---|---|---|
| | ***10%*** | ***20%*** | ***30%*** | ***10%*** | ***20%*** | ***30%*** |
| 1 year | 6.730 402 | 8.126 094 | 9.055 209 | 5.252 770 | 5.946 614 | 6.346 494 |
| 3 months | 6.741 903 | 9.164 045 | 6.416 011 | 5.319 278 | 6.892 313 | 5.123 868 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 7 days | 4.639 215 | 5.507 050 | 4.976 727 | 4.149 764 | 4.689 940 | 4.041 676 |

Table II presents the results for hourly data with block missingness. Compared to scattered missingness, block missingness results in significantly higher RMSE and MAE values, reflecting the difficulty of interpolating consecutive missing values.

For instance, for 1 year of data at a 10% missingness rate, the RMSE is 6.7304 ℃, compared to 1.9277 ℃ for scattered missingness under the same conditions. At a 30% missingness rate, the RMSE increases further to 9.0552 ℃. The MAE values show a similar trend, increasing from 5.2528 ℃ at 10% missingness to 6.3465 ℃ at 30%.

For shorter time spans, such as 7 days, the errors remain high but slightly lower than for the longer spans. For example, the RMSE for 7 days at 10% missingness is 4.6392 ℃, while the MAE is 4.1498 ℃. This difference underscores the increased difficulty of handling block missingness in longer time series, where patterns of missing data span larger intervals.

The results emphasize that while cubic spline interpolation can handle scattered missingness effectively, it struggles with block missingness, particularly in hourly data.

TABLE III. INTERPOLATION ERRORS FOR DAILY DATA WITH SCATTERED MISSINGNESS

| **Granularity** | **RMSE (℃)** | | | **MAE (℃)** | | |
|---|---|---|---|---|---|---|
| | ***10%*** | ***20%*** | ***30%*** | ***10%*** | ***20%*** | ***30%*** |
| 1 year | 0.479 914 | 0.529 246 | 0.771 659 | 0.397 070 | 0.381 967 | 0.566 188 |
| 3 months | 0.663 787 | 0.367 881 | 0.740 189 | 0.490 391 | 0.307 397 | 0.612 538 |

Table III summarizes the performance of cubic spline interpolation for daily data with scattered missingness. Compared to hourly data, the errors are significantly lower, reflecting the reduced complexity of interpolation at a coarser granularity.

For a 1-year time span, the RMSE for 10% missingness is only 0.4799 ℃, rising to 0.7717 ℃ at a 30% missingness rate. The corresponding MAE values are 0.3971 ℃ and 0.5662 ℃. For shorter time spans, such as 3 months, the RMSE and MAE values follow a similar pattern, with errors slightly higher for higher missingness rates.

These results demonstrate that reducing the data granularity simplifies the interpolation task, as the cubic spline method effectively captures the broader trends in daily data without overfitting to finer-scale variations.

TABLE IV. INTERPOLATION ERRORS FOR DAILY DATA WITH BLOCK MISSINGNESS

| **Granularity** | **RMSE (℃)** | | | **MAE (℃)** | | |
|---|---|---|---|---|---|---|
| | ***10%*** | ***20%*** | ***30%*** | ***10%*** | ***20%*** | ***30%*** |
| 1 year | 0.679 802 | 1.900 691 | 0.965 083 | 0.563 758 | 1.543 932 | 0.850 583 |
| 3 months | 0.460 572 | 1.192 308 | 1.069 782 | 0.383 953 | 0.915 439 | 0.899 323 |

Table IV reports the results for daily data with block missingness. As observed with hourly data, block missingness leads to higher errors compared to scattered missingness. However, the errors for daily data remain lower than those for hourly data.

For a 1-year time span at a 10% missingness rate, the RMSE is 0.6798 ℃, compared to 6.7304 ℃ for hourly data under the same conditions. At a 30% missingness rate, the RMSE is 0.9651 ℃, while the MAE is 0.8506 ℃. For shorter time spans, such as 3 months, the RMSE for 20% missingness is 1.1923 ℃, with a corresponding MAE of 0.9154 ℃.

These results highlight the robustness of cubic spline interpolation in handling block missingness at coarser temporal granularities. The lower errors compared to hourly data suggest that coarser data reduces the interpolation difficulty, even for challenging block missingness patterns.

## VII. CONCLUSION

This paper assessed the performance of cubic spline interpolation for imputing temperature data under two missingness patterns–scattered and block–across different temporal granularities and missingness rates. The results demonstrated that interpolation accuracy is highly influenced by the pattern, rate, and granularity of the missing data.

Scattered missingness yielded better interpolation accuracy compared to block missingness, where consecutive gaps posed greater challenges for imputation. Higher missingness rates led to greater interpolation errors with daily data consistently showing lower errors than hourly data due to its smoother nature and reduced variability.

The findings highlight the limitations of cubic spline interpolation in handling block missingness and emphasize the need for more robust methods when dealing with high missingness rates or block patterns. Future work could explore advanced imputation techniques, including machine learning approaches, to address these challenges and further improve accuracy.

## VII. APPENDIX

The source code for this paper is available on GitHub: https://github.com/iammadsfq/Cubic-Spline-Interpolation-for-Weather-Temperature-Data-Imputation

The video explaining this paper is available on YouTube: https://youtu.be/746-PJMFAXc

## VIII. Acknowledgment

Author expresses gratitude to all parties who have assisted in the preparation of this paper, especially to:

1. The Almighty God, for His grace and guidance, allowing the smooth completion of this paper.
2. Both parents, for providing both moral and material support to the author.
3. Extended family and friends who have encouraged and aided in the completion of this paper.
4. Dr. Judhi Santoso, M.Sc. and Arrival Dwi Sentosa, S.Kom, M.T. as the lecturers for the IF2123 Linear Algebra and Geometry course, for kindly imparting additional knowledge and offering solutions to challenges encountered in writing this paper.
5. Colleagues from the School of Electrical Engineering, especially those in the IF K3 class, for their support and collaborative spirit that have inspired and motivated the author throughout his journey.

The author deeply appreciates all the assistance, encouragement, and kindness received from these individuals and groups, without which the completion of this paper would not have been possible.

## References

[1] R. J. Little and D. B. Rubin, *Statistical Analysis with Missing Data,* 3rd ed., Wiley, 2020.

[2] Rorres, Chris and Howard Anton, *Applications of Linear Algebra* 3rd. 1984 New York: John Wiley and Sons.

[3] M. Ahmed and F. Haider, "Spline-based methods for temperature data reconstruction," *Journal of Environmental Monitoring*, vol. 15, no. 6, pp. 1327–1334, 2019.

[4] C. de Boor, *A Practical Guide to Splines*, Revised ed., Springer, 2001.

## Statement

I hereby declare that this paper I have written is my own work, not an adaptation or translation of someone else's paper, and not plagiarism.

Bandung, 1 Januari 2024

Ahmad Syafiq
13523135