

**Course Code: CCE-312**

**Course Title: Numerical Methods Sessional**

**A project Report On: A Comprehensive study of Factorization Method For wheather Alert System**

***Submitted To:***

**Dr. Md. Samsuzzaman**

**Professor**

Department Of Computer & Communication Engineering

Faculty of Computer Science & Engineering

***Submitted By:***

***Tasnim Ahammed Sohan***

Student ID: 2002061

Registration No: 09578

Level-03 semester-01

Session: 2020-21

Faculty of Computer Science and Engineering

**Patuakhali Science and Technology University**

**Dumki,Patuakhali-8602**

Date of Submission: 031-07-2024

**Table of Index**

Cover Page…………………………………………………………………………………..01

Abstract………………………………………………………………………………………03

Introduction………………………………………………………………………………….04

Objectives……………………………………………………………………………………05

Theory & Methodology……………………………………………………………………...05

Technology used…………………………………………………………………………….06

Features………………………………………………………………………………………07

Implementation………………………………………………………………………………08

User Guide……………………………………………………………………………………09

Source Code………………………………………………………………………………….12

Conclusion……………………………………………………………………………………18

References……………………………………………………………………………………18

Acknowledgements………………………………………………………………………….18

**Abstract**

**The "Weather Alert Model Using Cholesky Decomposition" project is a Python-based application designed to predict weather conditions and generate alerts for extreme temperatures in multiple regions. Utilizing Cholesky decomposition, a computationally efficient method for solving systems of linear equations, the model predicts temperatures based on simulated weather data represented by a positive-definite matrix. The project features a user-friendly graphical interface built with Tkinter, allowing users to interactively load data, generate predictions, and visualize results through intuitive bar charts. Additionally, the application issues alerts for regions exceeding predefined temperature thresholds, demonstrating the practical application of mathematical modeling in weather forecasting. This project highlights the integration of numerical methods, data analysis, and visualization, offering a valuable tool for educational and practical use in weather analysis and prediction systems**

**Introduction**

**Weather prediction plays a vital role in various domains, from agriculture and disaster management to daily life planning. Accurate and timely forecasting can help mitigate risks and improve preparedness for extreme weather conditions. The "Weather Alert Model Using Cholesky Decomposition" is a Python-based project designed to simulate weather data, predict regional temperatures, and provide alerts for extreme conditions.**

**The project leverages Cholesky decomposition, an efficient mathematical technique for solving systems of linear equations, to compute temperature predictions for multiple regions. Simulated weather data is represented using a symmetric positive-definite matrix, ensuring numerical stability and efficient computation.**

**A graphical user interface (GUI) built with Tkinter makes the application interactive and user-friendly. Users can load weather data files, generate predictions, and visualize results through bar charts. The model also features an alert system that notifies users of regions with temperatures exceeding a predefined threshold, demonstrating a practical application of numerical methods in real-world scenarios.**

**This project integrates mathematical modeling, data analysis, and visualization to showcase the utility of computational tools in weather forecasting. It serves as an educational tool to understand the interplay of numerical methods and programming while also providing insights into weather prediction systems.**

**Objectives**

**The primary objective of the "Weather Alert Model Using Cholesky Decomposition" project is to develop a Python-based application that efficiently predicts regional temperatures and provides alerts for extreme weather conditions. The key objectives include:**

1. **Implement Cholesky Decomposition: Utilize Cholesky decomposition to solve systems of linear equations for accurate temperature predictions.**
2. **Simulate Weather Data: Generate and analyze synthetic weather data for multiple regions.**
3. **Develop a User-Friendly GUI: Create an intuitive interface using Tkinter for seamless interaction with the application.**
4. **Visualize Predictions: Display predicted temperatures through bar charts for clear understanding and insights.**
5. **Issue Alerts: Identify and notify users of regions exceeding predefined temperature thresholds.**
6. **Integrate Data Handling: Allow users to load weather data from external files for flexible analysis.**

**This project aims to showcase the integration of mathematical modeling, programming, and data visualization in solving real-world problems like weather forecasting.**

**Theory And Methodology**

**Theory:**

**This project is based on the application of Cholesky decomposition, a numerical method used for solving systems of linear equations A×x=bA \times x = bA×x=b, where AAA is a symmetric positive-definite matrix. This decomposition breaks AAA into the product of a lower triangular matrix LLL and its transpose LTL^TLT such that A=L×LTA = L \times L^TA=L×LT. The advantage of this method is its computational efficiency and numerical stability, making it ideal for applications involving large datasets.**

**In this project:**

* **The matrix AAA represents the relationships between regions in the weather system.**
* **The vector bbb contains the initial temperature data for each region.**
* **The solution vector xxx provides the predicted temperatures for each region.**

**By leveraging Cholesky decomposition, the model efficiently predicts weather conditions, ensuring accuracy and speed.**

**Methodology**

1. **Weather Data Simulation:**
   * **A symmetric positive-definite matrix AAA is generated using random data to simulate inter-regional relationships.**
   * **A vector bbb is generated with random values to represent initial regional temperatures.**
2. **Cholesky Decomposition:**
   * **Decompose AAA into LLL and LTL^TLT.**
   * **Solve L×y=bL \times y = bL×y=b using forward substitution to find yyy.**
   * **Solve LT×x=yL^T \times x = yLT×x=y using backward substitution to find xxx, the predicted temperatures.**
3. **Alert System:**
   * **Define a temperature threshold (e.g., 30°C).**
   * **Compare predicted temperatures with the threshold and generate alerts for regions exceeding it.**
4. **Visualization:**
   * **Use Matplotlib to create a bar chart representing predicted temperatures by region.**
   * **Include a clear, user-friendly design for visual insights.**
5. **GUI Implementation:**
   * **Build the application interface using Tkinter.**
   * **Add functionality for:**
     + **Loading weather data from external CSV files.**
     + **Displaying predictions and alerts.**
     + **Visualizing results through an integrated chart.**
6. **Testing and Iteration:**
   * **Test the model with different datasets.**
   * **Validate the alert system and ensure accurate temperature predictions.**

**By combining Cholesky decomposition, data visualization, and a user-friendly GUI, the project effectively demonstrates how numerical methods can be applied to solve practical problems like weather forecasting.**

**Technology Used**

**The "Weather Alert Model Using Cholesky Decomposition" project utilizes the following technologies and tools:**

**Programming Language**

* **Python 3.11.9: The core programming language used for implementing the mathematical model, data visualization, and GUI development.**

**Libraries and Frameworks**

1. **NumPy:**
   * **Used for numerical computations, generating random data, and matrix operations.**
2. **SciPy:**
   * **Specifically, the scipy.linalg module is used to implement Cholesky decomposition.**
3. **Pandas:**
   * **Used for data manipulation, particularly for reading and processing weather data from CSV files.**
4. **Matplotlib:**
   * **Utilized for visualizing the predicted temperatures as bar charts.**
5. **Tkinter:**
   * **Used to build the graphical user interface (GUI), enabling interaction with the application.**

**Development Tools**

* **Python IDE/Text Editor: Any Python-compatible development environment such as PyCharm, VS Code, or Jupyter Notebook can be used to develop and test the application.**

**Operating System**

* **The project is platform-independent and can run on Windows, macOS, or Linux systems where Python 3.11.9 and the required libraries are installed.**

**By integrating these technologies, the project combines numerical computation, data handling, and user interactivity in a seamless manner.**

Features

The "Weather Alert Model Using Cholesky Decomposition" project includes the following key features:

1.Efficient Weather Prediction:

* Implement Cholesky Decomposition for solving linear systems, providing accurate and computationally efficient prediction of predictions of regional temperatures

2.Simulated Weather Data:

* Generates realistic weather data using random values to simulate inter-regional relationships and temperatures

3. Data Loading:

* Supports loading external weather data files CSV format, allowing flexibility in input data.

4. **Interactive Graphical User Interface (GUI)**:

* Build with Tkinter, the GUI provides a user-friendly platform to interact with the model
* Includes buttons for generating predictions and loading data

5. Alerts for Extreme Temperatures:

* Identifies regions with predicted temperatures exceeding a predefined threshold(e.g,30 c)
* Display alerts as warning to notify users about critical conditions

6. Data Visualization:

* Display predictions as a bar chart using Matplotlib for a clear and intuitive representation of regional temperatures

7. Dynamic And Scalable:

* Allow real-time prediction generation and visualization for a variable number of regions

8. Error Handling :

* Includes robust error handling mechanism to manage issues such as incorrect file formats or missing data

9. Educational Value:

* Demonstrate the application of Cholesky decomposition in real-world problems
* Offset insights into the integration of mathematical modeling, programming and data visualization

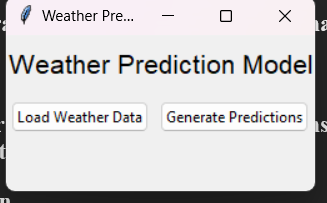
These features make the project both practical and educational, providing a valuable tool for understanding weather prediction and numerical methods

Implementation

**1. Weather Data Simulation**

The model simulates weather data for multiple regions, represented as a symmetric positive-definite matrix A and a vector b:

* **Matrix A**: Simulates inter-regional relationships, ensuring it is symmetric and positive-definite.
* **Vector b**: Represents random temperatures for regions.



**2. Cholesky Decomposition**

**Cholesky decomposition solves the system A\* x=b**

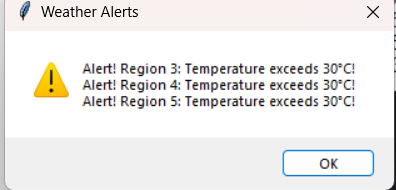
* **Decompose A into L (lower triangular matrix).**
* **Use forward substitution to solve L\*y=b.**
* **Use backward substitution to solve LT\*x=y.**

**3.Data Visualization**

**The predicted temperatures are visualized as a bar chart for easy interpretation**

**4.Weather Alerts**

**The system checks for extreme temperature conditions and generates alerts for regions exceeding a specified threshold.**

****

**5.GUI Implementation**

**The application uses Tkinter to create a user-friendly graphical interface:**

 Users can load weather data files or generate predictions.

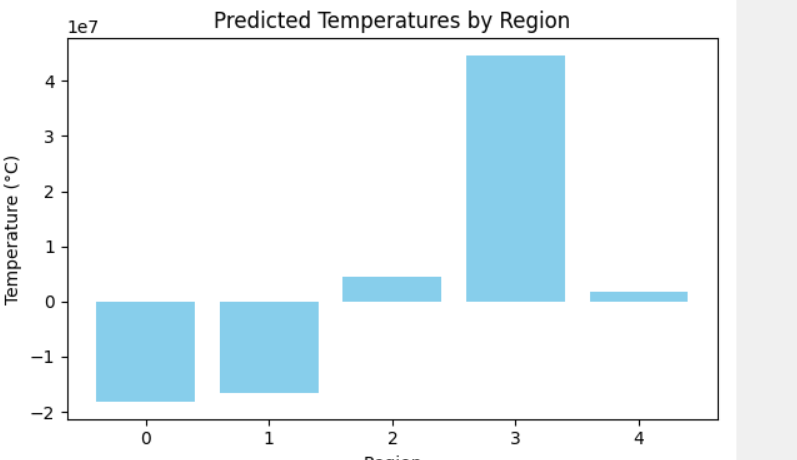
 Results and alerts are displayed in the interface.

 A bar chart is embedded to visualize predictions.

6. Data Loading and Error Handling

Users can load weather data from CSV files. Error handling ensures the program manages invalid files gracefully.

7.Prediction and Visualization Update

When predictions are generated, they are displayed in the GUI along with any weather alerts. 

**User Guide**

**This user guide explains how to operate the "Weather Alert Model Using Cholesky Decomposition" application, from setup to generating weather predictions and understanding the output.**

**1.System Requirement:**

**To run the project, ensure the following prerequisites are met:**

**Hardware Requirements**

* **A computer with at least 4GB of RAM.**
* **Windows, macOS, or Linux operating system**

**Software Requirements**

* **Python 3.11.9 installed**
* **Required Python libraries: numpy, scipy, pandas, matplotlib, and tkinter.  
  Install these by running**

**pip install numpy scipy pandas matplotlib**

**2.Running the Application**

**1. Download or Clone the Project**

* **Obtain the project code and ensure all files are in a single directory.**

**2.** **Run the Application**

* **Navigate to the project directory and run the main script**

**python <script\_name>.py**

* **Replace <script\_name> with the actual filename of the Python script (e.g., weather\_model.py).**

**3.** **Launch the GUI**

* **The application opens a graphical interface (GUI) where you can interact with the model.**

**3. Using The Application**

**Using the Application**

**Step 1: Load Weather Data (Optional)**

* **Click the "Load Weather Data" button to load a CSV file containing weather data.**
* **The file should be formatted as a table with temperature-related information.  
  If no file is loaded, the application will use simulated data instead.**

**Step 2: Generate Predictions**

* **Click the "Generate Predictions" button to calculate predicted temperatures for each region:**
  + **The predictions are based on simulated or loaded data.**
  + **Results are displayed in the form of region-wise temperature values.**

**Step 3: View Results and Alerts**

* **The application shows predicted temperatures for each region in the GUI.**
* **If any region’s temperature exceeds the threshold (default: 30°C), an alert will pop up, warning the user.**

**Step 4: Visualize Predictions**

* **A bar chart is displayed in the GUI showing predicted temperatures by region.**
* **The visualization helps users quickly identify regions with extreme temperatures.**

**4.Output details**

** Predicted Temperatures: Displayed as a list in the GUI, showing the calculated temperature for each region.**

** Alerts: Warning messages highlight regions where temperatures exceed the threshold.**

** Bar Chart: Graphical representation of regional temperatures for better interpretation.**

**5.Troubleshooting**

**Troubleshooting**

1. **Application Won't Start:**
   * **Ensure Python 3.11.9 and required libraries are installed.**
   * **Check for typos in the command used to run the script.**
2. **Error Loading Data:**
   * **Verify the CSV file is correctly formatted and accessible.**
3. **Unexpected Results:**
   * **Ensure data used (if loaded) is valid and aligns with the model's requirements**
4. **Tips for Users**

 To experiment with different datasets, create a CSV file with temperature data and load it using the GUI.

 Adjust the number of regions by modifying the n\_regions parameter in the generate\_weather\_data() function.

 Customize the temperature threshold for alerts by changing the high\_temp\_threshold variable in the check\_alerts() function.

7.Closing the Application

* **To close the application, simply exit the GUI by clicking the close button on the window or pressing Alt + F4 (Windows) or Command + W (macOS).**

**This user guide ensures that users can easily operate the application and understand its features, making it accessible for both technical and non-technical audiences.**

**Python Source code**

**import numpy as np**

**from scipy.linalg import cholesky**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**import tkinter as tk**

**from tkinter import ttk, filedialog, messagebox**

**from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg**

**def generate\_weather\_data(n\_regions=5):**

**"""Simulate weather data for multiple regions."""**

**np.random.seed(42)**

**A = np.random.rand(n\_regions, n\_regions)**

**A = A @ A.T  # Make A symmetric positive-definite**

**b = np.random.rand(n\_regions, 1) \* 30  # Random temperatures (0-30°C)**

**return A, b**

**def solve\_weather\_model(A, b):**

**"""Solve the system using Cholesky decomposition."""**

**L = cholesky(A, lower=True)**

**y = np.linalg.solve(L, b)  # Forward substitution**

**x = np.linalg.solve(L.T, y)  # Back substitution**

**return x.flatten()**

**def plot\_weather\_data(predictions):**

**"""Plot the predicted weather data."""**

**fig, ax = plt.subplots(figsize=(6, 4))**

**ax.bar(range(len(predictions)), predictions, color='skyblue')**

**ax.set\_xlabel('Region')**

**ax.set\_ylabel('Temperature (°C)')**

**ax.set\_title('Predicted Temperatures by Region')**

**plt.tight\_layout()**

**return fig**

**def open\_file():**

**"""Open a dialog to select weather data and load it."""**

**file\_path = filedialog.askopenfilename(**

**title="Select Weather Data File",**

**filetypes=[("CSV Files", "\*.csv"), ("All Files", "\*.\*")]**

**)**

**if file\_path:**

**try:**

**data = pd.read\_csv(file\_path)**

**messagebox.showinfo("File Loaded", f"Data loaded from: {file\_path}")**

**print(data.head())  # Print a preview of the data**

**except Exception as e:**

**messagebox.showerror("Error", f"Failed to load file: {e}")**

**else:**

**messagebox.showinfo("No File", "No file selected.")**

**def check\_alerts(predictions):**

**"""Check for temperature alerts and return alert messages."""**

**alerts = []**

**high\_temp\_threshold = 30  # Temperature threshold for alert**

**for i, temp in enumerate(predictions):**

**if temp > high\_temp\_threshold:**

**alerts.append(f"Alert! Region {i + 1}: Temperature exceeds {high\_temp\_threshold}°C!")**

**return alerts**

**def update\_predictions():**

**"""Generate and display predictions."""**

**A, b = generate\_weather\_data(n\_regions=5)**

**predictions = solve\_weather\_model(A, b)**

**# Update label with predictions**

**predictions\_text = "\n".join([f"Region {i + 1}: {temp:.2f}°C"**

**for i, temp in enumerate(predictions)])**

**result\_label.config(text=predictions\_text)**

**# Check for weather alerts**

**alerts = check\_alerts(predictions)**

**if alerts:**

**alert\_message = "\n".join(alerts)**

**messagebox.showwarning("Weather Alerts", alert\_message)**

**# Plot the results in the GUI**

**fig = plot\_weather\_data(predictions)**

**canvas = FigureCanvasTkAgg(fig, master=root)**

**canvas.draw()**

**canvas.get\_tk\_widget().grid(row=3, column=0, columnspan=2, pady=10)**

**def create\_gui():**

**"""Create the Tkinter GUI for the weather model."""**

**global root, result\_label**

**root = tk.Tk()**

**root.title("Weather Prediction Model Using Cholesky Decomposition")**

**# Widgets**

**ttk.Label(root, text="Weather Prediction Model", font=("Arial", 16)).grid(row=0, column=0, columnspan=2, pady=10)**

**ttk.Button(root, text="Load Weather Data", command=open\_file).grid(row=1, column=0, pady=5)**

**ttk.Button(root, text="Generate Predictions", command=update\_predictions).grid(row=1, column=1, pady=5)**

**result\_label = ttk.Label(root, text="", font=("Arial", 12), anchor="w", justify="left")**

**result\_label.grid(row=2, column=0, columnspan=2, pady=10)**

**# Start the Tkinter event loop**

**root.mainloop()**

**if \_\_name\_\_ == "\_\_main\_\_":**

**create\_gui()**

**Conclusion**

**The Weather Alert Model Using Cholesky Decomposition project effectively combines numerical methods and user-friendly design to predict regional temperatures. By using Cholesky decomposition, it efficiently solves systems of equations to forecast weather data, and the Tkinter GUI allows users to interact with the model easily. The system provides real-time alerts for extreme temperatures and visualizes predictions through bar charts. This project demonstrates the practical use of mathematical modeling for weather prediction and can be expanded with real-world data for further applications**

**References**

* **NumPy Documentation:** [**https://numpy.org**](https://numpy.org)
* **SciPy Documentation:** [**https://scipy.org**](https://scipy.org)
* **Matplotlib Documentation:** [**https://matplotlib.org**](https://matplotlib.org)
* **Tkinter Documentation:** [**https://docs.python.org/3/library/tkinter.html**](https://docs.python.org/3/library/tkinter.html)

**Acknowledgement**

**I would like to express my sincere gratitude to my project supervisor for their guidance and support throughout this project. Special thanks to the open-source community for tools like SciPy, Matplotlib, Tkinter, and Pandas, which were essential in implementing the model. I also appreciate the contributions of the Cholesky decomposition algorithm’s developers. Lastly, I would like to thank my friends and family for their continuous encouragement.**