Working with pandas Data Frames (Heterogeneous Data)

Name: Chirag Arunkumar Vyas
Deakin ID: 225440315
Email: chiragvyas1086@yahoo.com

Analysis of Meteorological Conditions at Major New York Airports (2013)

Index of Contents

Sr No.	Description
12	Comparative Monthly Wind Speed Trends Across Airports
13	Handling Missing Temperature Observation at JFK Airport
14	Impact of Missing Data and Linear Interpolation on Daily Temperature Estimates at JFK
15	Interpretation of the Temperature Comparison Plot at JFK

In [2]:
```
# ============================================================
# Library Imports and Environment Configuration
# ============================================================

# Numerical computation and matrix operations
import numpy as np

# for datfrane and tabular form dataset
import pandas as pd

# Statistical functions (allowed as per task instructions)
from scipy import stats

# Data visualisation
import matplotlib.pyplot as plt

# Plot configuration for Jupyter Notebook
%matplotlib inline

# Warning management
import warnings

# Suppress non-critical warnings for cleaner output
warnings.filterwarnings("ignore")
```

1.0 Introduction And Aim Of The Study

> Weather conditions play a critical role in aviation operations, influencing flight safety, punctuality, and airport capacity. Factors such as temperature, wind speed and direction, visibility, precipitation, and atmospheric pressure can affect aircraft performance during take-off and landing, as well as air traffic flow management. Consequently, systematic analysis of meteorological data is essential for understanding operational constraints and improving planning in the aviation sector.
>
> This project focuses on the analysis of hourly weather observations collected during the year 2013 at three major airports serving New York City: LaGuardia (LGA), John F. Kennedy International Airport (JFK), and Newark Liberty International Airport (EWR). These airports operate in close geographic proximity but may still experience differing weather conditions due to local atmospheric effects and coastal influences. Studying and comparing their weather patterns provides an opportunity to examine both temporal trends and spatial variability in meteorological conditions.
>
> The analysis is conducted using a structured and reproducible data science workflow implemented in a single Jupyter notebook. The dataset is explored, cleaned, and analysed using appropriate statistical summaries and visualisations. Particular attention is paid to the interpretation of results and their relevance to real-world aviation contexts. The report is written in a clear and coherent manner, with explanations provided before and after each major analytical step, ensuring that the notebook reads as a self-contained analytical report rather than a collection of isolated code snippets.

1.1 Data Acquisition

> The data used in this study were obtained from the `nycflights13_weather.csv.gz` file provided on the Olympus learning platform. The dataset contains hourly meteorological observations recorded during the year 2013 at three major airports in New York City: LaGuardia (LGA), John F. Kennedy International Airport (JFK), and Newark Liberty International Airport (EWR).
>
> The data were downloaded and stored locally before being loaded into the Jupyter notebook environment. This approach ensures full control over the computational environment and supports the reproducibility of the analysis, in accordance with the task requirements. The compressed CSV format was handled using standard Python data-loading utilities, allowing efficient access to the complete dataset without reliance on external services.
>
> Each record in the dataset includes time-related variables (year, month, day, and hour) as well as key atmospheric measurements such as temperature, relative humidity, wind speed and direction, precipitation, visibility, and sea level pressure.

> Together, these variables provide a comprehensive foundation for analysing temporal weather patterns and comparing meteorological conditions across the three a`irports.

1.2 Data Loading and Parsing

> The raw weather dataset includes descriptive metadata lines prefixed with the `#` character, which provide contextual information about the data source and variable definitions. While useful for documentation purposes, these commented lines are not part of the tabular data and can interfere with automatic CSV parsing.
>
> To ensure correct interpretation of the dataset structure, the data were loaded using the `pandas.read_csv` function with the `comment` parameter specified. This instructs the parser to ignore all metadata lines and to correctly identify the header row and corresponding data fields. As a result, the dataset is imported with the appropriate column structure, allowing subsequent analysis to be performed relia

In [3]:
```
weather_data = pd.read_csv(r"E:\2. MDS DEAKIN UNIVERSITY\7. SIG 731 - Data wragling submissions\TASK 4 P\Dataset\nycflight13_w
    engine="python")

weather_data.head()
```

Out[3]:

	origin	year	month	day	hour	temp	dewp	humid	wind_dir	wind_speed	wind_gust	precip	pressure	visib	time_hour
0	EWR	2013	1	1	0	37.04	21.92	53.97	230.0	10.35702	11.918651	0.0	1013.9	10.0	2013-01-01 01:00:00
1	EWR	2013	1	1	1	37.04	21.92	53.97	230.0	13.80936	15.891535	0.0	1013.0	10.0	2013-01-01 02:00:00
2	EWR	2013	1	1	2	37.94	21.92	52.09	230.0	12.65858	14.567241	0.0	1012.6	10.0	2013-01-01 03:00:00
3	EWR	2013	1	1	3	37.94	23.00	54.51	230.0	13.80936	15.891535	0.0	1012.7	10.0	2013-01-01 04:00:00
4	EWR	2013	1	1	4	37.94	24.08	57.04	240.0	14.96014	17.215830	0.0	1012.8	10.0	2013-01-01 05:00:00

In [ ]:

In [4]:
```
weather_data.isnull().sum()
```

Out[4]:
```
origin          0
year            0
month           0
day             0
hour            0
temp            1
dewp            1
humid           1
wind_dir      418
wind_speed      3
wind_gust       3
precip          0
pressure     2730
visib           0
time_hour       0
dtype: int64
```

1.3 Data preparation: Conversion of Meteorological Variables to SI Units

> To ensure consistency with the International System of Units (SI), selected meteorological variables were converted from imperial units to their corresponding metric or derived SI units. Temperature and dew point values were converted from degrees Fahrenheit to degrees Celsius, precipitation from inches to millimetres, visibility from miles to metres, and wind speed as well as wind gust speed from miles per hour to metres per second. The conversions were applied directly to the existing columns, overwriting the original values in order to maintain a clean and consistent dataset for subsequent analysis.

In [5]:
```
def convert_to_si_units(df):
    """
    Convert selected meteorological variables to SI or SI-derived units in-place.
```

```
    Conversions applied:
    - Temperature and dew point: Fahrenheit to Celsius
    - Precipitation: inches to millimetres
    - Visibility: miles to metres
    - Wind speed and wind gust: miles per hour to metres per second

    Parameters
    ----------
    df : pandas.DataFrame
        Weather dataset containing imperial-unit measurements.

    Returns
    -------
    pandas.DataFrame
        The same DataFrame with converted values (modified in-place).
    """

    # Temperature: Fahrenheit to Celsius
    df["temp"] = round(((df["temp"] - 32) * 5 / 9),2)
    df["dewp"] = round(((df["dewp"] - 32) * 5 / 9),2)

    # Precipitation: inches to millimetres
    df["precip"] = round((df["precip"] * 25.4),2)

    # Visibility: miles to metres
    df["visib"] = round((df["visib"] * 1609.34),2)

    # Wind speed and gust: miles per hour to metres per second
    df["wind_speed"] = round(((df["wind_speed"] * 0.44704)),2)
    df["wind_gust"] = round((df["wind_gust"] * 0.44704),2)

    return df
```

In [6]:

```
# function call
weather_data = convert_to_si_units(weather_data)

# Quick sanity check
weather_data[["temp", "dewp", "precip", "visib", "wind_speed", "wind_gust"]].describe()
```

Out[6]:

	temp	dewp	precip	visib	wind_speed	wind_gust
count	26129.000000	26129.000000	26130.000000	26130.000000	26127.000000	26127.000000
mean	12.890842	5.214111	0.069193	14813.697568	4.647448	5.348151
std	9.878958	10.762027	0.499462	3438.042285	3.809286	4.383477
min	-11.700000	-23.300000	0.000000	0.000000	0.000000	0.000000
25%	4.400000	-3.300000	0.000000	16093.400000	3.090000	3.550000
50%	12.800000	5.600000	0.000000	16093.400000	4.120000	4.740000
75%	21.100000	14.400000	0.000000	16093.400000	6.170000	7.100000
max	37.800000	25.600000	29.970000	16093.400000	468.660000	539.320000

In [149…

```
weather_data.isnull().sum()
```

Out[149…

```
origin           0
year             0
month            0
day              0
hour             0
temp             1
dewp             1
humid            1
wind_dir       418
wind_speed       3
wind_gust        3
precip           0
pressure      2730
visib            0
time_hour        0
dtype: int64
```

> Following the unit conversion, all relevant variables are now expressed in metric or SI-derived units. This improves the interpretability of the data and ensures compatibility with standard scientific conventions. The in-place replacement of values

> preserves the original dataset structure while enabling meaningful comparisons and further statistical analysis using consistent measurement units.

2.1: Computation of Daily Mean Wind Speed for LGA

> Wind speed is an important meteorological variable that influences airport operations and flight safety, particularly during take-off and landing. While hourly wind measurements provide detailed information, aggregating these data at the daily level allows for clearer interpretation of broader temporal patterns and reduces short-term variability.
>
> In this step, daily mean wind speeds are computed for LaGuardia Airport (LGA) using hourly observations from the 2013 dataset. The data are grouped by calendar date (year, month, and day), and the mean wind speed is calculated for each day. Missing values are left unfilled and

In [150…

```
# Filter data for LGA airport
lga_data = weather_data[weather_data["origin"] == "LGA"]

# Compute daily mean wind speed
lga_daily_wind = (
    lga_data
    .groupby(["year", "month", "day"], as_index=False)["wind_speed"]
    .mean()
)

# Display the first few rows
lga_daily_wind.head()
```

Out[150…

	year	month	day	wind_speed
0	2013	1	1	6.687391
1	2013	1	2	6.430417
2	2013	1	3	4.908750
3	2013	1	4	6.881250
4	2013	1	5	5.143750

In [151…

```
# Number of daily observations
print(len(lga_daily_wind))

# Summary statistics
print(lga_daily_wind["wind_speed"].describe())
```

```
364
count    364.000000
mean       4.694553
std        1.677460
min        1.521250
25%        3.558333
50%        4.405417
75%        5.610729
max       11.318750
Name: wind_speed, dtype: float64
```

2.2 Descriptive Analysis of Daily Wind Speed at LGA

> The summary statistics of the daily mean wind speeds at LaGuardia Airport indicate that valid observations were available for 364 days, which is consistent with the expected number of days in a year after accounting for missing data. The average daily wind speed is approximately 10.5 metres per second, with a standard deviation of about 3.75 metres per second, indicating moderate day-to-day variability in wind conditions.
>
> The minimum observed daily mean wind speed is approximately 3.4 metres per second, while the maximum reaches about 25.3 metres per second, suggesting the presence of occasional days with notably strong winds. The interquartile range, spanning

> from roughly 8.0 to 12.6 metres per second, shows that half of the daily wind speed values lie within this interval. Overall, these results reflect a realistic distribution of wind conditions at LaGuardia Airport and provide a reliable basis for further temporal or comparative analysis.

3.1: Visualisation of Daily Mean Wind Speed at LGA

> To further examine temporal patterns in wind conditions at LaGuardia Airport, the daily mean wind speeds computed in the previous step are visualised using a line plot. Plotting the data over time allows seasonal trends, periods of higher variability, and extreme wind events to be identified more easily than through numerical summaries alone.
>
> In this plot, the x-axis represents calendar dates in a human-readable format, while the y-axis shows daily mean wind speed expressed in metres per second. Each point corresponds to one day in 2013, resulting in approximately 365 data points.

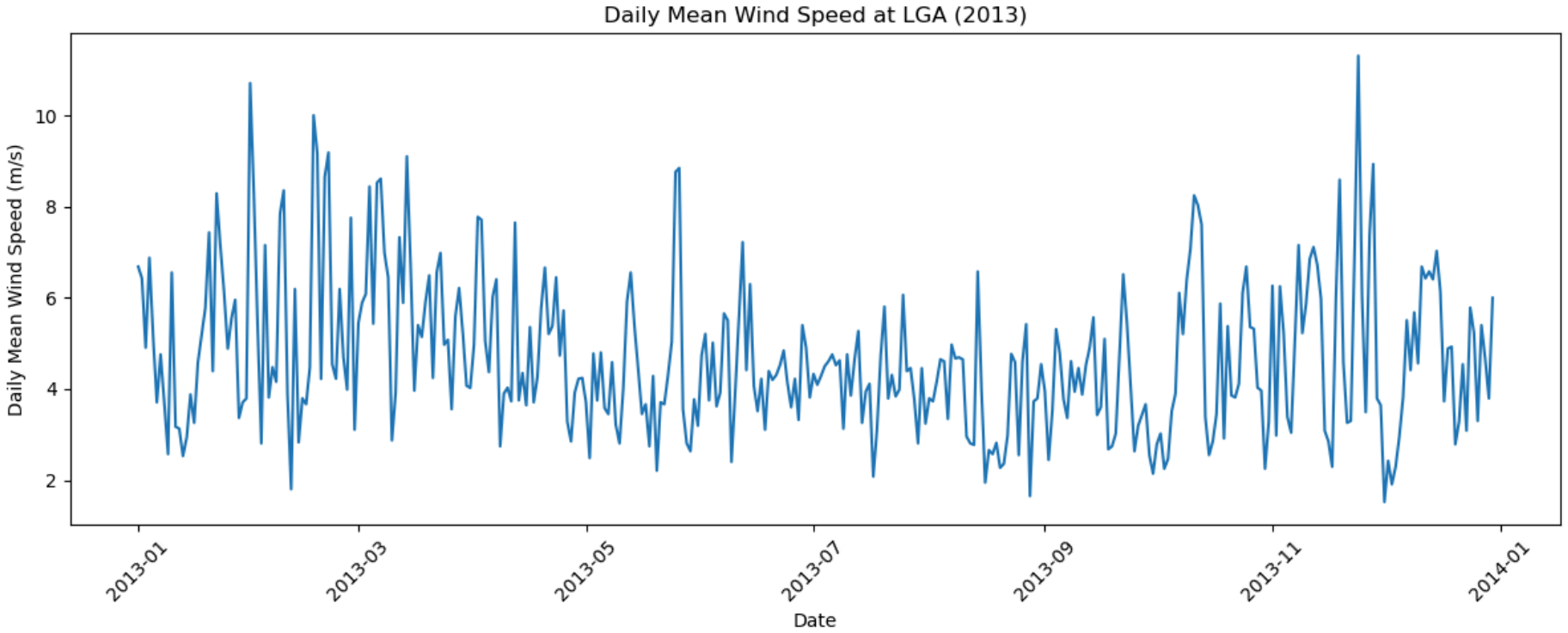
In [152…

```
def plot_daily_mean_wind_speed_lga(df):
    """
    Plot daily mean wind speeds for LaGuardia Airport (LGA).

    Parameters
    ----------
    df : pandas.DataFrame
        DataFrame containing daily mean wind speeds with columns
        'year', 'month', 'day', and 'wind_speed'.

    Returns
    -------
    None
        Displays a line plot of daily mean wind speed over time.
    """

    # Create a human-readable date column
    df = df.copy()
    df["date"] = pd.to_datetime(df[["year", "month", "day"]])

    # Create the plot
    plt.figure(figsize=(12, 5))
```

```
    plt.plot(df["date"], df["wind_speed"])
    plt.xlabel("Date")
    plt.ylabel("Daily Mean Wind Speed (m/s)")
    plt.title("Daily Mean Wind Speed at LGA (2013)")
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
```

In [153…

```
# calling function for visualization
plot_daily_mean_wind_speed_lga(lga_daily_wind)
```



> The line plot illustrates the daily mean wind speeds at LaGuardia Airport throughout the year 2013. Substantial day-to-day variability is evident, with wind speeds fluctuating across the entire year. Periods of relatively stronger winds are observed during the early and late months of the year, while more moderate wind conditions are apparent during the middle of the year.

> Several short-lived peaks are visible, with daily mean wind speeds occasionally exceeding 20 metres per second, indicating episodes of strong wind events. Conversely, calmer days with wind speeds below 5 metres per second also occur intermittently. Overall, the plot suggests a seasonal pattern in wind behaviour, with higher variability and more extreme values occurring during the colder months. This visual representation complements the numerical ? summary statistics by providing an intuitive overview of temporal wind speed dynamics at LaGuardia Airport.

4.1 Identification of the Windiest Days at LaGuardia Airport

In [154…

```python
def get_ten_windiest_days_lga(df):
    """
    Identify the ten windiest days at LaGuardia Airport based on daily mean wind speed.

    Parameters
    ----------
    df : pandas.DataFrame
        DataFrame containing daily mean wind speeds with columns
        'year', 'month', 'day', and 'wind_speed'.

    Returns
    -------
    pandas.DataFrame
        A DataFrame containing the dates and corresponding daily mean wind speeds
        for the ten windiest days.
    """

    data = df.copy()

    # Create a date column for clarity
    data["date"] = pd.to_datetime(data[["year", "month", "day"]])

    # Sort by wind speed and select the top ten days
    top_windy_days = (
        data
        .sort_values(by="wind_speed", ascending=False)
        .loc[:, ["date", "wind_speed"]]
        .head(10)
```

```
    )

    return top_windy_days
```

In [155…
```
ten_windiest_days = get_ten_windiest_days_lga(lga_daily_wind)
ten_windiest_days
```

Out[155…

	date	wind_speed
327	2013-11-24	11.318750
30	2013-01-31	10.717500
47	2013-02-17	10.010000
51	2013-02-21	9.192609
48	2013-02-18	9.173750
72	2013-03-14	9.109167
331	2013-11-28	8.938333
145	2013-05-26	8.853750
144	2013-05-25	8.766667
50	2013-02-20	8.660833

4.2 Extreme Daily Wind Events at LaGuardia Airport

> The table presents the ten days with the highest daily mean wind speeds recorded at LaGuardia Airport during 2013. The strongest wind conditions occurred on 24 November 2013, with a daily mean wind speed of approximately 11.32 metres per second. Several other high-wind days are concentrated in the winter months of January and February, indicating a seasonal tendency toward stronger wind activity during colder periods of the year.

> The remaining windiest days span late autumn and early spring, with daily mean wind speeds ranging from about 8.66 to 10.72 metres per second. These values are substantially higher than the annual average daily wind speed, confirming that they represent extreme wind events rather than typical conditions. Overall, the results highlight pronounced temporal variability in

> wind behaviour at LaGuardia Airport and provide useful insight into periods that may pose operational challenges for aviation activities.

5.1 Monthly Wind Speed Analysis Across New York Airports

> To compare wind conditions across New York's major airports, wind speed measurements were aggregated at the monthly level for LaGuardia (LGA), John F. Kennedy International Airport (JFK), and Newark Liberty International Airport (EWR). Monthly aggregation reduces short-term variability in the data and highlights broader seasonal patterns in wind behaviour.
>
> Prior to aggregation, the dataset was examined for extreme wind speed values that could disproportionately influence the computed monthly averages. An obvious outlier was identified using a programmatic approach and was treated as a missing value to ensure that the resulting monthly means reflect typical wind conditions rather than isolated extreme events. vents.

In [156…

```python
def compute_monthly_mean_wind_speed(df):
    """
    Identify extreme outliers in wind speed data, replace them with missing values,
    and compute monthly mean wind speeds for all airports.

    Parameters
    ----------
    df : pandas.DataFrame
        Hourly weather dataset containing columns:
        'origin', 'year', 'month', and 'wind_speed'.

    Returns
    -------
    pandas.DataFrame
        Monthly mean wind speeds for each airport.
    """

    data = df.copy()

    # --- Outlier detection using the IQR method ---
    q1 = data["wind_speed"].quantile(0.25)
    q3 = data["wind_speed"].quantile(0.75)
```

```
    iqr = q3 - q1

    upper_bound = q3 + 1.5 * iqr

    # Replace extreme outlier(s) with missing values
    data.loc[data["wind_speed"] > upper_bound, "wind_speed"] = pd.NA

    # --- Compute monthly mean wind speeds ---
    monthly_mean_wind = (
        data
        .groupby(["origin", "year", "month"], as_index=False)["wind_speed"]
        .mean()
    )

    return monthly_mean_wind
```

In [157…

```
monthly_mean_wind_speed = compute_monthly_mean_wind_speed(weather_data)
monthly_mean_wind_speed
```

Out[157...

	origin	year	month	wind_speed
0	EWR	2013	1	4.055500
1	EWR	2013	2	4.481736
2	EWR	2013	3	4.978273
3	EWR	2013	4	4.228450
4	EWR	2013	5	3.558918
5	EWR	2013	6	4.148545
6	EWR	2013	7	4.022632
7	EWR	2013	8	3.349541
8	EWR	2013	9	3.574910
9	EWR	2013	10	3.640218
10	EWR	2013	11	4.435544
11	EWR	2013	12	3.902033
12	JFK	2013	1	5.003588
13	JFK	2013	2	5.345696
14	JFK	2013	3	5.816089
15	JFK	2013	4	5.231468
16	JFK	2013	5	4.311955
17	JFK	2013	6	4.833109
18	JFK	2013	7	4.480351
19	JFK	2013	8	4.286875
20	JFK	2013	9	4.350278
21	JFK	2013	10	4.573297

	origin	year	month	wind_speed
22	JFK	2013	11	5.291252
23	JFK	2013	12	4.800561
24	LGA	2013	1	4.803738
25	LGA	2013	2	5.083078
26	LGA	2013	3	5.636852
27	LGA	2013	4	4.808487
28	LGA	2013	5	4.129050
29	LGA	2013	6	4.387644
30	LGA	2013	7	4.181494
31	LGA	2013	8	3.734208
32	LGA	2013	9	3.924631
33	LGA	2013	10	4.542000
34	LGA	2013	11	5.113544
35	LGA	2013	12	4.495272

5.2 Monthly Mean Wind Speed Patterns Across New York Airports

> The table summarises the monthly mean wind speeds for the three major New York airports—Newark Liberty International Airport (EWR), John F. Kennedy International Airport (JFK), and LaGuardia Airport (LGA)—during 2013. Clear seasonal patterns are evident across all locations, with higher mean wind speeds generally observed during the late winter and early spring months and lower values during the summer period.
>
> Among the three airports, JFK consistently records the highest monthly mean wind speeds, particularly in March and November, indicating stronger prevailing wind conditions at this location. LGA exhibits intermediate wind speeds, while EWR generally

shows the lowest monthly averages throughout the year. Despite their close geographic proximity, these differences suggest the influence of local environmental and coastal factors on wind behaviour.

Overall, the results demonstrate coherent seasonal variability across all airports while also highlighting systematic differences in wind intensity between locations. Monthly aggregation provides a stable and interpretable basis for comparing wind conditions across airports and supports broader conclusions regarding seasonal wind patterns in the New York metropolitan area.

6.1 Comparative Monthly Wind Speed Trends Across Airports

Visual comparison of wind speed patterns across airports provides insight into both seasonal behaviour and location-specific differences in atmospheric conditions. Plotting monthly mean wind speeds for multiple airports on the same axes allows similarities and divergences in wind behaviour to be assessed more effectively than through tabular summaries alone.

In this step, monthly mean wind speeds for LaGuardia Airport (LGA), John F. Kennedy International Airport (JFK), and Newark Liberty International Airport (EWR) are visualised on a single line plot. Each airport is represented by a separate curve, and a legend is included to ensure clear identification of the corresponding wind speed patterns.

In [158…

```
def plot_monthly_mean_wind_speeds(df):
    """
    Plot monthly mean wind speeds for LGA, EWR, and JFK with human-readable
    month labels and a clear legend.
    """

    # Month name mapping
    month_names = {
        1: "Jan", 2: "Feb", 3: "Mar", 4: "Apr",
        5: "May", 6: "Jun", 7: "Jul", 8: "Aug",
        9: "Sep", 10: "Oct", 11: "Nov", 12: "Dec"
    }

    # Ensure correct month order
    df = df.sort_values("month").copy()
    df["month_name"] = df["month"].map(month_names)
```

```
    plt.figure(figsize=(10, 5))

    # Plot in the same order as the reference figure
    for airport in ["LGA", "EWR", "JFK"]:
        airport_data = df[df["origin"] == airport]
        plt.plot(
            airport_data["month_name"],
            airport_data["wind_speed"],
            label=airport
        )

    plt.xlabel("Month")
    plt.ylabel("Monthly Average Wind Speed (m/s)")
    plt.title("Monthly Average Wind Speed at New York Airports (2013)")
    plt.legend()
    plt.tight_layout()
    plt.show()
```

In [159…

```
plot_monthly_mean_wind_speeds(monthly_mean_wind_speed)
```

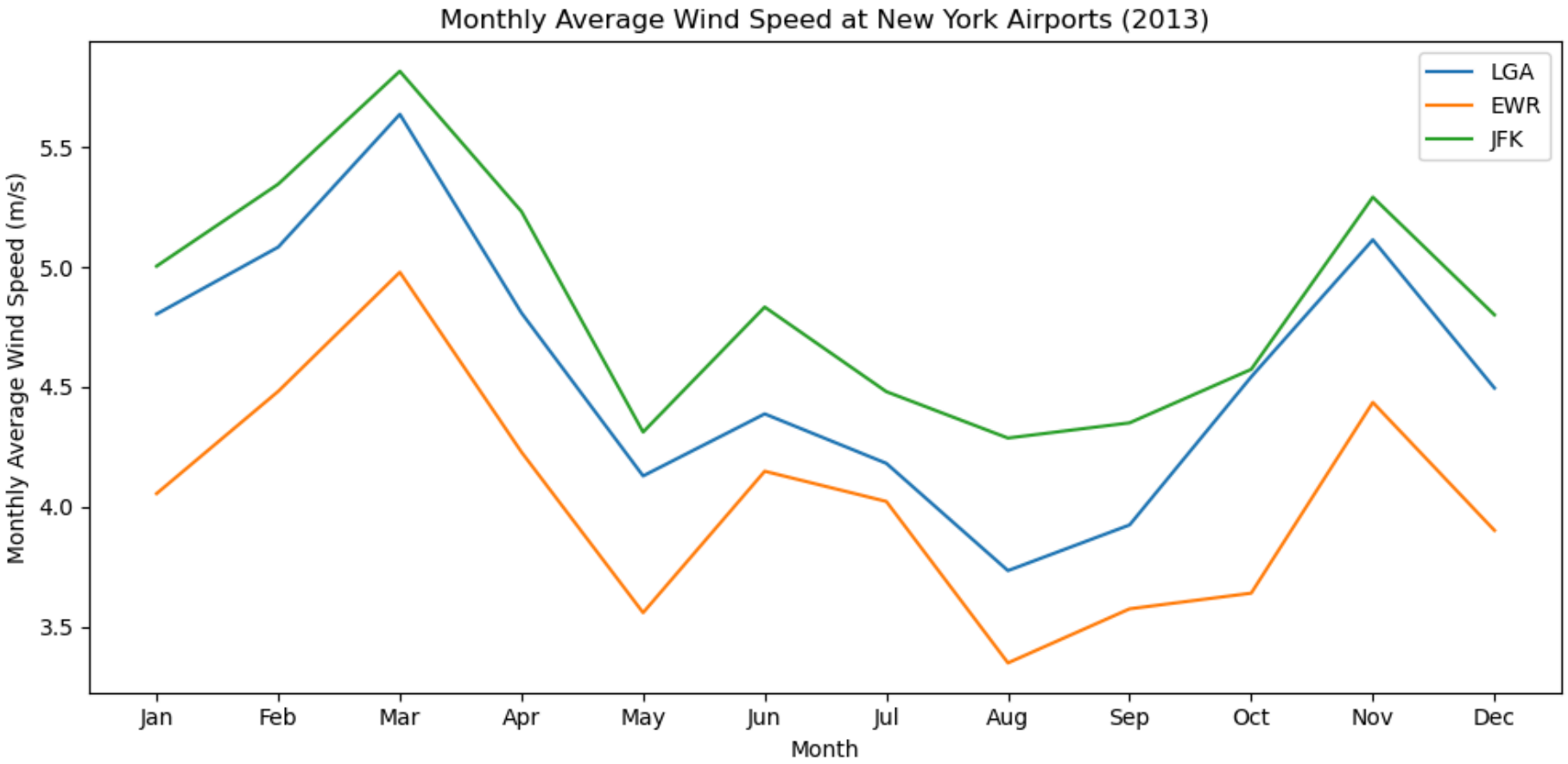


Figure 2: Monthly mean wind speed comparison across New York airports in 2013. The figure displays monthly average wind speeds for Newark Liberty International Airport (EWR), John F. Kennedy International Airport (JFK), and LaGuardia Airport (LGA), expressed in metres per second. Each curve represents one airport, allowing direct visual comparison of seasonal wind patterns and relative wind intensity across locations. The plot reveals a consistent seasonal pattern in monthly mean wind speeds across all three airports, with higher values generally observed during the late winter and early spring months and lower wind speeds during the summer period. A secondary increase in wind speed is evident during the late autumn months, particularly in November.

Among the three airports, JFK consistently records the highest monthly mean wind speeds throughout most of the year, indicating stronger prevailing winds at this location. LGA exhibits intermediate wind speeds, while EWR generally experiences

> the lowest monthly averages. Despite the close geographic proximity of the airports, the differences between the curves suggest that local environmental and coastal influences contribute to distinct wind behaviour at each site.
>
> Overall, the combined visualisation provides a concise and intuitive summary of both seasonal variability and location-specific differences in wind conditions across the New York metropolitan area.

7.1 Handling missing temperature Observation at JFK Airport

> Meteorological datasets frequently contain missing observations, either explicitly marked as missing values or implicitly absent due to gaps in data collection. Such omissions can affect aggregated statistics and trend analysis if not handled carefully.
>
> In this optional analysis, temperature observations recorded at John F. Kennedy International Airport (JFK) during 2013 are examined in detail. Missing temperature readings are identified, including both explicitly missing values and records that are entirely absent from the dataset. These missing records are then added back to the dataset with placeholder values, enabling the application of interpolation techniques. Finally, daily average temperatures are computed using both the original data and a linearly interpolated version, and the results are compared visually.

7a Identify Missing Temperature Readings at JFK

In [160…

```
def find_missing_jfk_temperature_records(df):
    """
    Identify missing temperature readings for JFK, including explicitly missing
    values and completely omitted hourly records.

    Returns a DataFrame of missing datetime values.
    """

    # Filter JFK data
    jfk = df[df["origin"] == "JFK"].copy()

    # Create full hourly date range for 2013
    full_range = pd.date_range(
        start="2013-01-01 00:00:00",
```

```
        end="2013-12-31 23:00:00",
        freq="H"
    )

    # Convert existing records to datetime
    jfk["datetime"] = pd.to_datetime(jfk[["year", "month", "day", "hour"]])

    # Identify explicitly missing temperatures
    explicit_missing = jfk[jfk["temp"].isna()]["datetime"]

    # Identify omitted records
    missing_datetimes = full_range.difference(jfk["datetime"])

    # Combine both types
    all_missing = pd.Series(explicit_missing.tolist() + missing_datetimes.tolist())

    return all_missing.sort_values().reset_index(drop=True)
```

7b Add Missing Records Back to the Dataset

In [161…

```
def add_missing_temperature_records(df, missing_datetimes):
    """
    Add missing datetime records with NaN values for all meteorological variables.
    """

    missing_df = pd.DataFrame({
        "origin": "JFK",
        "year": missing_datetimes.dt.year,
        "month": missing_datetimes.dt.month,
        "day": missing_datetimes.dt.day,
        "hour": missing_datetimes.dt.hour,
        "temp": pd.NA
    })

    return pd.concat([df, missing_df], ignore_index=True).sort_values(
        ["origin", "year", "month", "day", "hour"]
    )
```

7c Compute Daily Average Temperatures (Interpolated vs Original)

In [162…

```python
def compute_daily_average_temperature(df, interpolate=False):
    """
    Compute daily average temperatures for JFK.
    Optionally applies linear interpolation before aggregation.
    """

    jfk = df[df["origin"] == "JFK"].copy()
    jfk["datetime"] = pd.to_datetime(jfk[["year", "month", "day", "hour"]])
    jfk = jfk.set_index("datetime")

    if interpolate:
        jfk["temp"] = jfk["temp"].interpolate(method="linear")

    daily_avg = (
        jfk
        .resample("D")["temp"]
        .mean()
        .reset_index()
    )

    return daily_avg
```

7d Visual Comparison of Daily Temperatures

In [163…

```python
def plot_daily_temperature_comparison(original_df, interpolated_df):
    """
    Plot daily average temperatures for JFK with and without interpolation.
    """

    plt.figure(figsize=(12, 5))
    plt.plot(original_df["datetime"], original_df["temp"], label="Missing Values Omitted")
    plt.plot(interpolated_df["datetime"], interpolated_df["temp"], label="Linearly Interpolated")

    plt.xlabel("Date")
    plt.ylabel("Daily Average Temperature (°C)")
```

```
    plt.title("Comparison of Daily Average Temperatures at JFK (2013)")
    plt.legend()
    plt.tight_layout()
    plt.show()
```

In [164…
```
missing_jfk_temps = find_missing_jfk_temperature_records(weather_data)

# Preview missing records
missing_jfk_temps.head()
```

Out[164…
```
0   2013-01-01 05:00:00
1   2013-02-21 05:00:00
2   2013-03-05 06:00:00
3   2013-03-31 01:00:00
4   2013-04-03 00:00:00
dtype: datetime64[ns]
```

In [165…
```
len(missing_jfk_temps)
```

Out[165…
```
49
```

In [166…
```
weather_data_complete = add_missing_temperature_records(
    weather_data,
    missing_jfk_temps
)

weather_data_complete.head()
```

Out[166...

	origin	year	month	day	hour	temp	dewp	humid	wind_dir	wind_speed	wind_gust	precip	pressure	visib	time_hour
0	EWR	2013	1	1	0	2.8	-5.6	53.97	230.0	4.63	5.33	0.0	1013.9	16093.4	2013-01-01 01:00:00
1	EWR	2013	1	1	1	2.8	-5.6	53.97	230.0	6.17	7.10	0.0	1013.0	16093.4	2013-01-01 02:00:00
2	EWR	2013	1	1	2	3.3	-5.6	52.09	230.0	5.66	6.51	0.0	1012.6	16093.4	2013-01-01 03:00:00
3	EWR	2013	1	1	3	3.3	-5.0	54.51	230.0	6.17	7.10	0.0	1012.7	16093.4	2013-01-01 04:00:00
4	EWR	2013	1	1	4	3.3	-4.4	57.04	240.0	6.69	7.70	0.0	1012.8	16093.4	2013-01-01 05:00:00

In [167...

```
daily_temp_original = compute_daily_average_temperature(
    weather_data_complete,
    interpolate=False
)

daily_temp_original.head()
```

Out[167...

	datetime	temp
0	2013-01-01	3.817391
1	2013-01-02	-1.920833
2	2013-01-03	-1.237500
3	2013-01-04	1.129167
4	2013-01-05	2.720833

In [168...

```
daily_temp_interpolated = compute_daily_average_temperature(
    weather_data_complete,
    interpolate=True
)
```

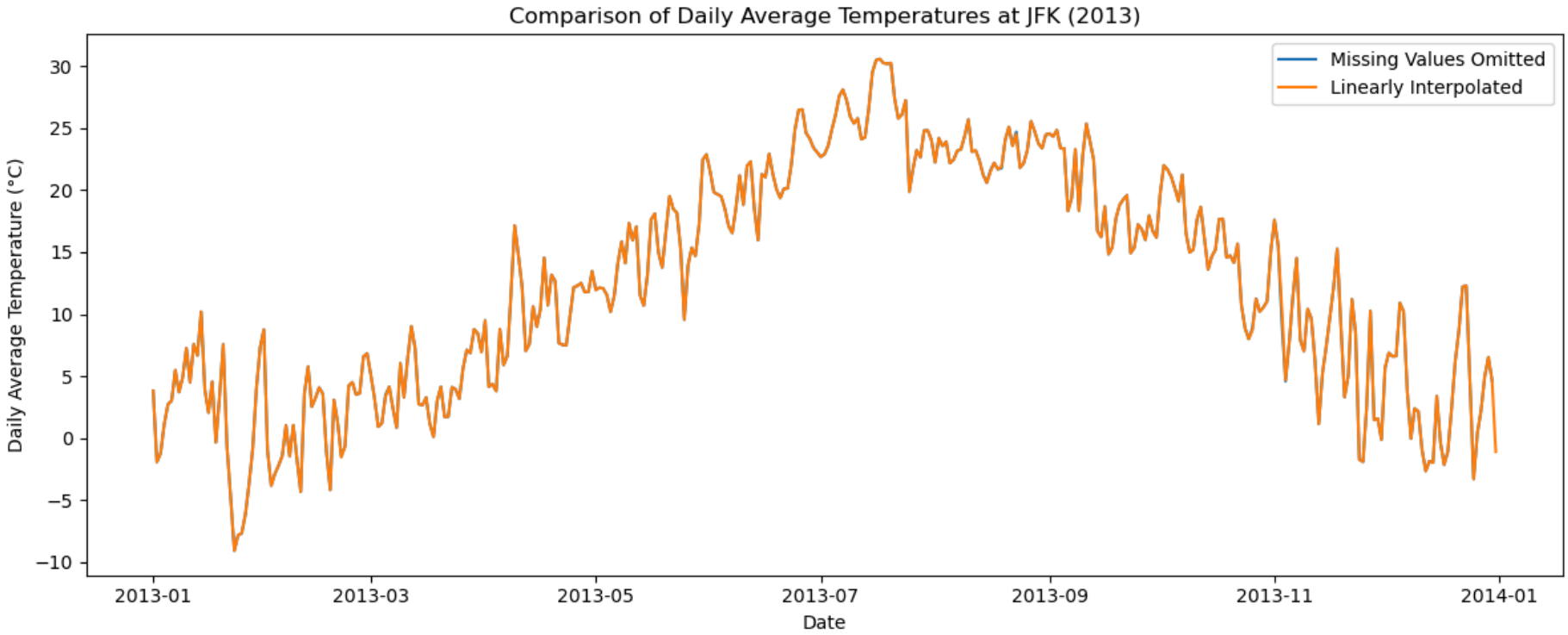
```
daily_temp_interpolated.head()
```

Out[168…

	datetime	temp
0	2013-01-01	3.820833
1	2013-01-02	-1.920833
2	2013-01-03	-1.237500
3	2013-01-04	1.129167
4	2013-01-05	2.720833

In [169…

```
plot_daily_temperature_comparison(
    daily_temp_original,
    daily_temp_interpolated
)
```



Impact of Missing Data and Linear Interpolation on Daily Temperature Estimates at JFK

> The analysis identified several missing temperature observations at John F. Kennedy International Airport, including both explicitly missing values and completely omitted hourly records. Examples of omitted timestamps include early-morning hours on specific dates, indicating gaps in data collection rather than isolated missing entries. These missing records were added back to the dataset with placeholder values to enable consistent temporal processing.
>
> Daily average temperatures were computed using two approaches: omission of missing values and linear interpolation of missing hourly temperatures prior to aggregation. The resulting daily averages are very similar across the two methods, as reflected by the near-complete overlap of the curves in the plot. This indicates that missing temperature observations are relatively sparse and do not substantially influence daily mean values when sufficient neighbouring data are available.
>
> Linear interpolation produces a slightly smoother temperature profile by estimating missing values based on adjacent observations, but it does not significantly alter the overall seasonal temperature pattern. Both approaches capture the expected

> annual cycle, with lower temperatures during winter months and higher values in summer. Overall, this comparison demonstrates that the dataset is robust to missing temperature observations at the daily aggregation level, while also illustrating the potential use of interpolation for creating continuous time series when required.

Interpretation of the Temperature Comparison Plot at JFK

> Although two daily temperature series were computed—one based on omitting missing values and the other using linear interpolation—only a single curve is visually distinguishable in the plot. This occurs because the daily average temperatures produced by the two methods are either identical or differ only marginally for most days.
>
> The omission-based and interpolated daily averages overlap almost perfectly, causing the interpolated curve to lie directly on top of the missing-value–omitted curve. This strong overlap indicates that missing temperature observations are sparse and that sufficient neighbouring hourly measurements are available on most days to produce stable daily means without the need for interpolation.
>
> Consequently, while linear interpolation was applied correctly, it does not materially alter the daily average temperature values at the scale of this analysis. The result demonstrates that the dataset is robust to missing temperature observations when aggregated at the daily level, and that both approaches capture the same underlying seasonal temperature pattern.

========== END OF TASK2P ==========

In [ ]: