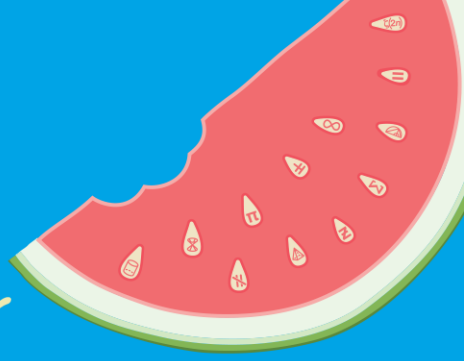
AMSI VACATIONRESEARCH SCHOLARSHIPS 2021–22

Get a taste for Research this Summer



Infilling Missing Data in Climate Time Series

Hanyi Wang

Supervised by John Boland and Sleiman Farah

University of South Australia

Vacation Research Scholarships are funded jointly by the Department of Education and Training and the Australian Mathematical Sciences Institute

AMSI AUSTRALIAN MATHEMATICAL SCIENCES INSTITUTE

Contents

1 Abstract

This project presents infilling methods of missing climate time series values considering three types of patterns of missing data, with both scattered and continuous gaps. Three time series variables, namely solar farm output, wind farm output and ambient temperature were studied. An additive model consisting of a Fourier series and an autoregressive model was applied to model and simulate the seasonal and stochastic variations.

2 Introduction

Detecting and handling missing data could be the essential pre-processing task in the time series modelling area. The occurrence of the missing values might be attributed to several reasons, e.g., faulty measuring instruments or human errors. In climate time series analysis, construction of forecast models depends on the quality of the data as parameter estimation is affected by the gaps [Ramos-Calzado et al. 2008].

Some conventional methods to deal with missing values such as simply deleting or replacing the gaps with mean values, could be applied for a small amount of missing values. However, the model results under this approach will be inaccurate or even biased for the increasing number of missing records [Pratama et al. 2016]. More advanced missing data imputation for the time series variables could be dealt with using interpolation e.g., linear or using cubic splines. However, for longer gaps of days or weeks, which is often the case in some climate variables, more sophisticated techniques will have to be employed.

In this project, we used Fourier series and autoregressive (AR) model [Farah & Boland 2021, Boland 2020] to infill missing data for solar farm output, wind farm output and ambient temperature. The results of ambient temperature are shown in the Appendix.

Statement of Authorship

John and Sleiman conceived the main idea and outline for the infilling methods, guided and supervised the project work, and proofread the report. John designed the proposal as well as provided the data. Under the continuing academical assistance from John and Sleiman, Hanyi designed and created the missing values, developed the code in R, built the models, produced the infilling outcome, and wrote this report. AMSI and the Australian Department

of Education funded the project.

3 Data

The data used in this project is kindly provided by my proposal supervisor, Professor John Boland, which is collected from two areas in Australia, i.e., New South Wales and South Australia (see Table 3.1). All variables are complete.

Two types of models are typically used to decompose time series data, additive and multiplicative. Equation 3.1 is the equation of additive time series and Equation 3.2 is the equation of multiplicative time series. In this project, we are using additive model in particular.

$$\texttt{Additive TS} = \texttt{Trend} + \texttt{Seasonality} + \texttt{Random Noise} \tag{3.1}$$

$$\texttt{Multiplicative TS} = \texttt{Trend} \times \texttt{Seasonality} \times \texttt{Random Noise} \tag{3.2}$$

Table 3.1 describes those variables with their locations and data characteristics. In this project, we consider the variables at both low-resolution levels such as daily, and high-resolution levels such as 5 and 30 minutes.

Table 3.1: Data Characteristics.

Variable	Location	Interval	Duration	Characteristics
Solar Farm Output	Broken Hill	5-minute	2 years	No Trend
Wind Farm Output	Snowtown	30-minute	15 years	No Trend No Seasonality
Maximum Temperature	Kent Town	Daily	43 years	No Trend

We simulate the scattered and continuous gaps among the variables such that the situations where data is missing are generated, and we assume the records were missing completely at random for simplicity, such that our models don't require additional algorithms for investigating the cause of missing patterns and the likely values for the gap [Moritz et al. 2015, Rubin 1976].

What's more, we use day as unit for data in high-resolution level, and year as unit for data in low-resolution level.

Three types of artificial gaps are studied. The first one is the case when the observations are missing within one day (or one year), the second one is the case when the observations are

missing more than one day (or one year), the last one is the case when there is a miscellaneous missing situation with both scattered and continuous gaps at random.

If there is more than 1/3 of values missing within one-day (or one-year) period, then we define it as a single day (year) missing. For solar farm output, a single day missing is defined by if more than 1/3 of values missing in the mid-day where the sunshine comes, such that we get rid of the meaningless zero values.

4 Models and Results

4.1 Methods

The first step is determining the type of time series, that is, detect whether the variable has inherent trend and seasonality, or with some random noise, and find whether to use additive decomposition or multiplicative decomposition. This could be reviewed in Table 3.1.

The way we infill the gaps is based on patterns of missing data and types of the time series for each variable, as shown in Figure 4.1.1.

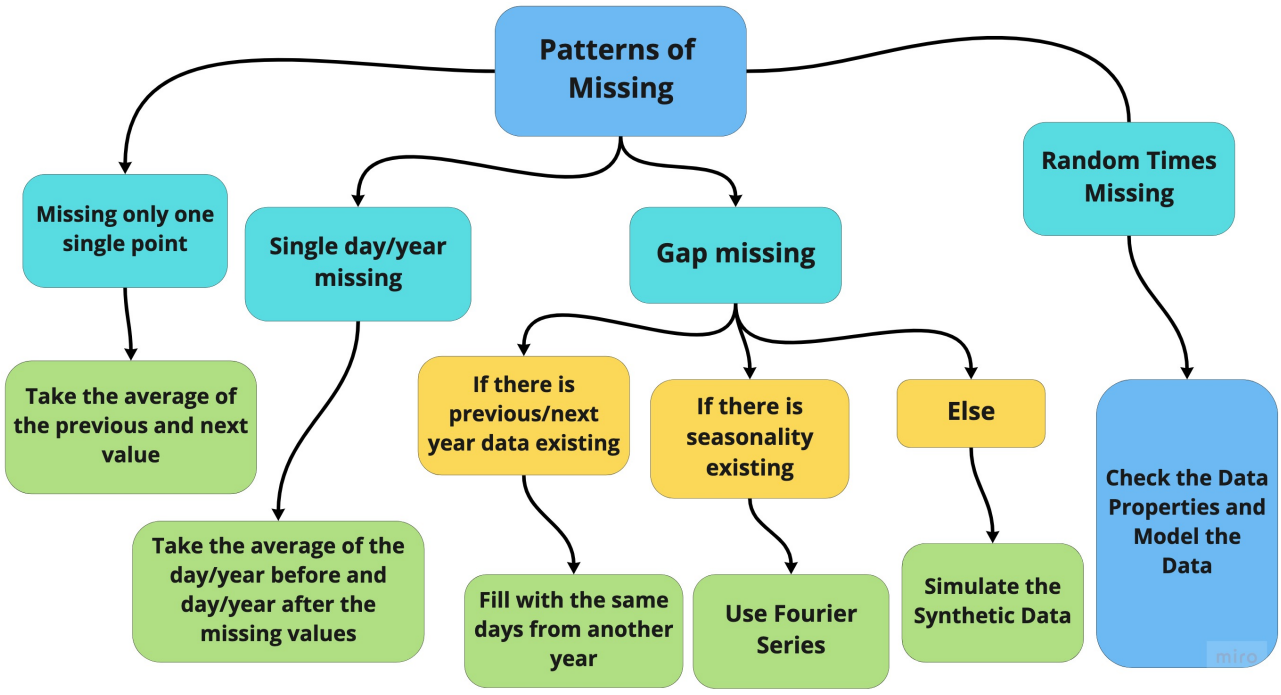


Figure 4.1.1: Roadmap of the methodology.

The easiest case is when one data record is missing, we could take the average of the previous

and next value to fill that missing place. If the data is missing for a whole day, then we simply take the average of the day before and day after the missing values.

If there is a gap missing, we have several solutions depending on the cases. If another year of data is available, we could just fill the gaps with the same days from this year. If there is no other year available, and the variable has seasonality, we could use the Fourier series model. Otherwise, we may need to simulate the synthetic data.

If data is missing at random times, we need to model the data and simulate the synthetic data according to the data characteristics. In summary, different modelling techniques are utilised to handle different parts of the series (see Figure 4.1.2).

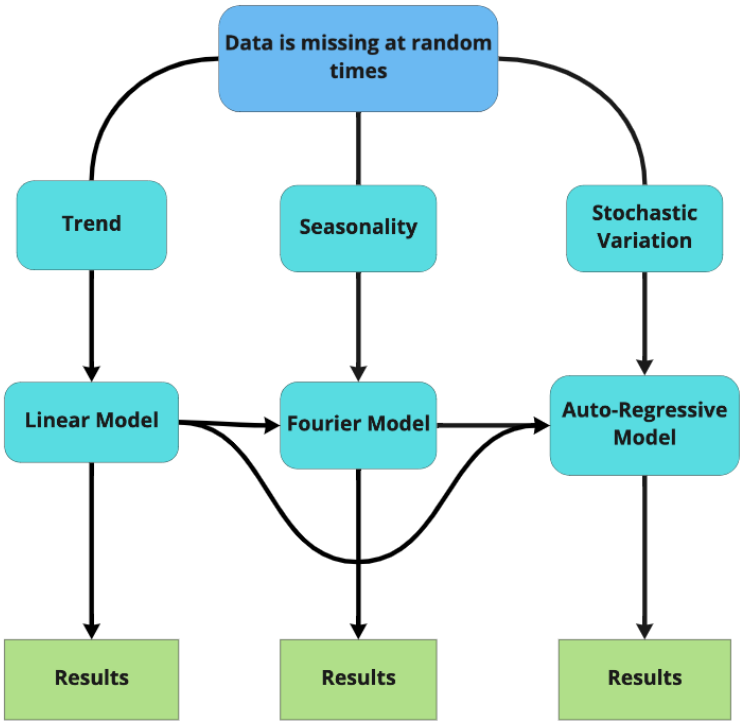


Figure 4.1.2: Roadmap of the methodology for miscellaneous missing situation at random.

For an additive time series which is the case in this project, the following forecast models are considered. We first detect the trend using a linear model with a result of fitted values `F1` and residuals `R1`. `R1` is simply the difference between the original data and fitted values. Then the seasonality of the data could be modelled using the Fourier series model on `R1`, and we obtain the new fitted values `F2` with its residuals `R2`. Apart from the trend and seasonality, autoregressive model can be used to model to the remaining stochastic variations which is `R2`

here using its autocorrelation. The last fitted values `F3` is calculated. The final result is the sum of all fitted values which is `F1 + F2 + F3`.

If there is no trend, we skip the linear model. If there is no seasonality, we skip the Fourier series model. Similarly, if there is no stochastic variation, then we skip the autoregressive model.

When we start to infill the missing values, Fourier series model is trained first from a complete series and then the same parameters are applied to the testing series; linear model and autoregressive model are used for generating the synthetic data for testing data directly. To validate our results, we could either look at the line plot or find whether our models did the work properly. For example, the residuals of the autoregressive model should be uncorrelated and their mean should be zero.

4.1.1 Fourier Series Model

To identify the significant seasonal component, one way is to use the power spectrum when we are not able to find the cycles clearly by looking at the graph. Power spectrum is a plot of the signal power lying at each frequency, the more significant cycles, the more the contribution to the variance of the series, the greater the power [Boland 2010].

As long as we find the most significant cycles during the sample interval n, we could minimize the sum of the squared deviations of the model from the data to find the optimal a_i and b_i using the equation as follows

$$F(t) = a_0 + \sum_{i}^{k} (a_i cos(2\pi it/n) + b_i sin(2\pi it/n)) \tag{4.1}$$

$$t = 1, 2, 3, \ldots, n,\ a_0 = avg(S_t),$$

$$k = \text{significant cycles during the sample interval } n$$

Consequently, power model $F(t)$ is able to be calculated using Equation 4.1, given a complete training series S_t.

4.1.2 Autoregressive Model and Synthetic Generation

An autoregressive model forecasts the target variable using a linear combination of past values of itself. The reason we did this is because of the autocorrelation existing within the time

series.

Autocorrelation describes the linear dependences between the current series C_t and its lagged version $C_{t-1}, C_{t-2}, \ldots, C_{t-k}$ therefore providing insights on how their degree of similarity and tendency evolves in time. As such, it is a key element in refining information from the data and filling in the gaps.

An autoregressive model of order p can be written as

$$C_t = \alpha_0 + \alpha_1 C_{t-1} + \alpha_2 C_{t-2} + \ldots + \alpha_p C_{t-p} + \varepsilon_t, \tag{4.2}$$

where $t = 1, 2, \ldots, n$, α_0 is the constant, $\alpha_1, \alpha_2, \ldots, \alpha_p$ is the coefficients of the AR model, $C_{t-1}, C_{t-2}, \ldots, C_{t-p}$ are lagged values of C_t, and ε_t is the remaining random noise. It is assumed that the ε_t is independent and identically distributed (i.i.d.).

Equation 4.2 is also used for the situation when we need to simulate the synthetic data. It happens when there is a large gap of missing data, that there is not enough existed lagged data to guide the further fluctuation appropriately. Therefore, we have to manually detect the missing places and fill those missing records after we perform the AR model.

This could be done using the form of Equation 4.2 and the AR model parameters we obtained previously. The procedure is shown in Algorithm 1. As a first approximation, a normal distribution is investigated to sample ε_t here, and this would be refined by working on the distributional characteristics in our further work.

Moreover, if there is trend in the series, then the constraint in Algorithm 1 will change to $MIN(S_t) < C_t + F(t) + Y(t) < MAX(S_t)$ is true, where $Y(t)$ is the fitted values we got when we model its trend.

Akaike information criterion (AIC) is used to find the optimal p order. Given a set of candidate models with different parameter, the preferred model is the one with the minimum AIC value.

AIC could be written as

$$AIC = T \ln\left(\frac{SSE}{T}\right) + 2(k+2),$$

where T is the number of observations used for estimation and k is the number of predictors in the model [Hyndman & Athanasopoulos 2021].

Algorithm 1: Filling the Missing Values using Fourier series plus Autoregressive Algorithm when the Series has no Trend.

Input: Incomplete C_t, Incomplete raw series S_t, Complete seasonal model fits $F(t)$

Output: Complete C_t

$M \leftarrow MAX(S_t)$;
$m \leftarrow MIN(S_t)$;
for $t = 1$ **to** n **do**
  if $C_t == null \wedge C_{t-1} != null \wedge \ldots \wedge C_{t-p} != null$ **then**
    $C_t \leftarrow \alpha_0 + \alpha_1 \times C_{t-1} + \ldots + \alpha_p \times C_{t-p}$;
    $C_t \leftarrow C_t + \varepsilon_t$; // where $\varepsilon_t \sim N(0, \sigma_t^2)$
    /* make sure $MIN(S_t) < C_t + F(t) < MAX(S_t)$ */
    while $C_t + F(t) > M \vee C_t + F(t) < m$ **do**
      $C_t \leftarrow \alpha_0 + \alpha_1 \times C_{t-1} + \ldots + \alpha_p \times C_{t-p} + \varepsilon_t$;
    end
  end
end
return C_t;

4.2 Infilling Results of Solar Farm Output

We take five-day data to train our model for solar farm output. There is no trend detected in the data, so we use power spectrum to find the significant cycles first. In the power spectrum, the significant spikes determine which frequencies are to be included in the Fourier series model.

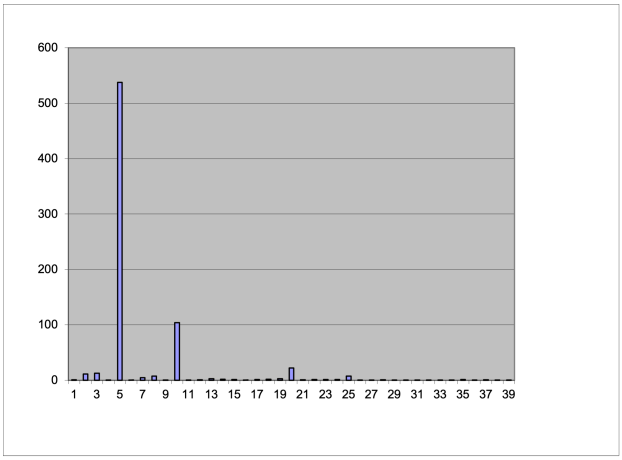


Figure 4.2.1: Solar Farm Output Power Spectrum.

Figure 4.2.1 demonstrates that there are 5, 10 and 20 cycles in 5 days. We note that these frequencies are reasonable because they match our knowledge for solar radiation; once-a-day distinguishes the days, twice-a-day and four-times-a-day point to the fluctuation of solar radiation within a day.

We define the Fourier series representation of the solar farm output series as follows:

$$F(t) = 14.5 + \sum_{i}^{\{5,10,20\}} \left(a_i cos(2i\pi t/1440) + b_i sin(2i\pi t/1440)\right), \quad (4.3)$$
$$t = 1, 2, 3, \ldots, 1440.$$

Table 4.1 lists the a_i, b_i values that minimise the sum of squared deviations of the model from the data.

Table 4.1: Values of the Fourier Series Model Parameters of Solar Farm Output.

i	a_i	b_i
5	−22.7960	−4.2534
10	9.4369	3.8729
20	−4.0137	−2.4325

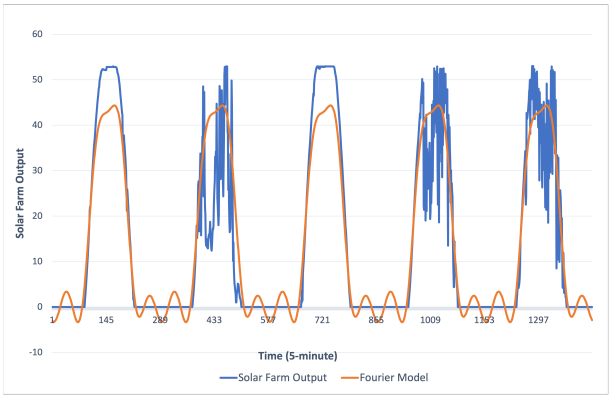
Figure 4.2.2 shows the model results we got from training set, it verifies that the summarised result of Fourier series model and AR(5) model is sufficient for modelling the solar farm output. Figure 4.2.3 is the result we got when we apply the same Fourier series model to testing set.

The optimal AR model for the testing residuals we got from Fourier series model is

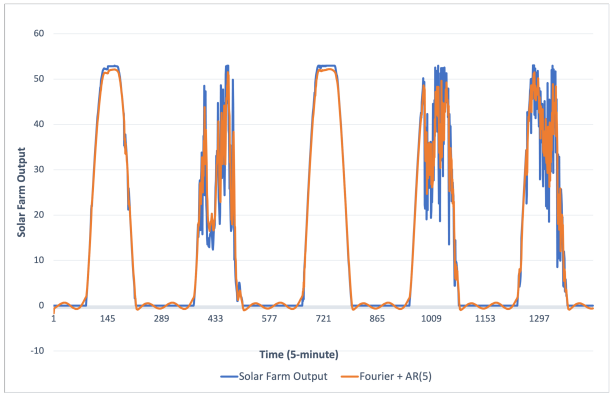
$$C_t = 0.6713 \times C_{t-1} - 0.0479 \times C_{t-2} + 0.1586 \times C_{t-3} + 0.0568 \times C_{t-4} + 0.1240 \times C_{t-5} + \varepsilon_t. \quad (4.4)$$

To be noted that equation of AR(5) for training set is different from Equation 4.4 as testing set has different autocorrelation from training set. We simulate the synthetic data using Equation 4.4 to fill the missing values in the case when the data is missing quite a lot.

Figure 4.2.4, Figure 4.2.5 and Figure 4.2.6 demonstrate the results we got for solar farm output. It could be seen in Figure 4.2.4, that our filled data perfectly covers the raw data which is great. In Figure 4.2.5, we use the data from previous year so the fluctuations are not similar, but it is still reasonable. Figure 4.2.6 illustrates the synthetic data we simulated. Our model result here is in between the minimum and maximum of the raw data which looks successful. Meanwhile, Figure 4.2.6 proves that different subsets of data from the same variable have their own AR models, and the presence of missing records do reduce the model quality.



(a) Fourier Series Model.



(b) Fourier Series Model + AR(5) Model.

Figure 4.2.2: Model Results of Solar Farm Output Training Set.

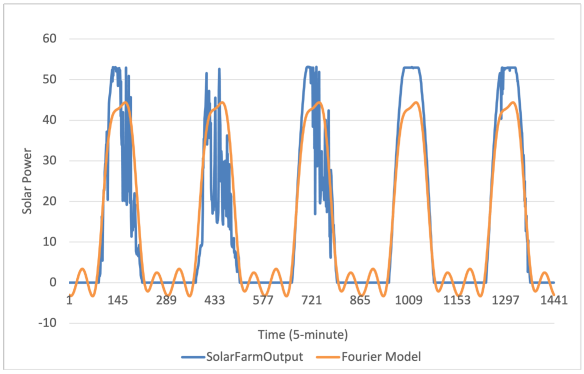
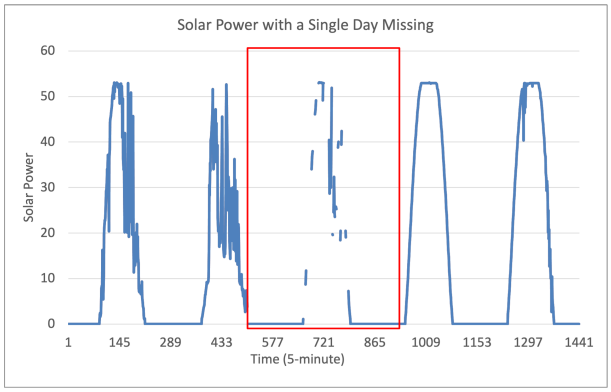
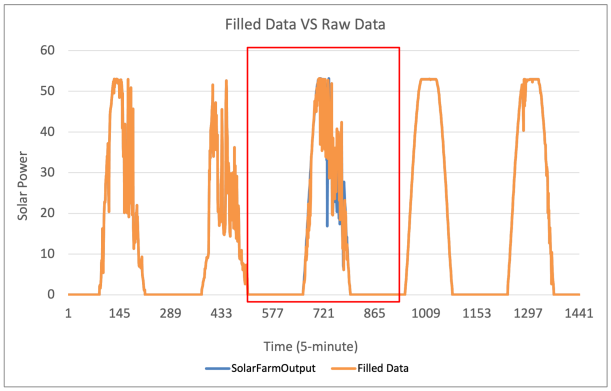


Figure 4.2.3: Fourier Series Model for Solar Farm Output Testing Set.

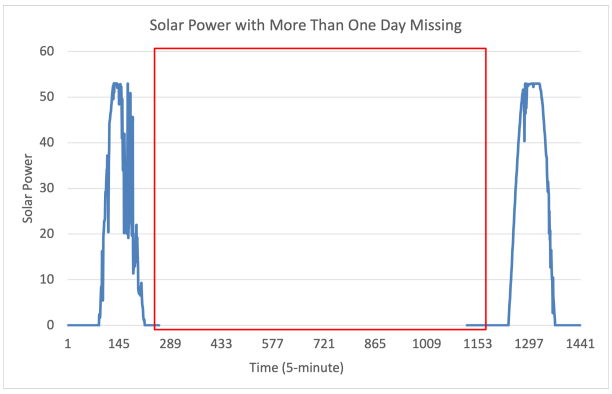


(a) Single Day Missing.

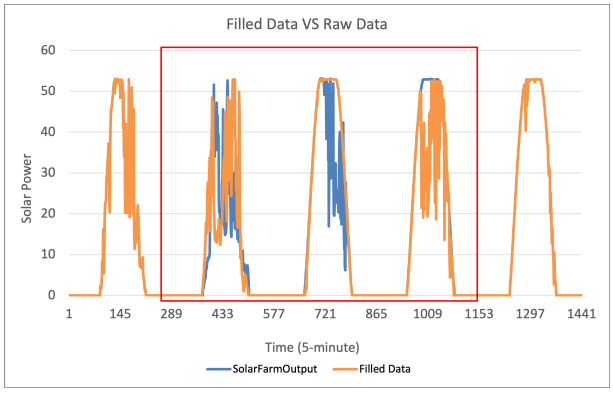


(b) Filled using Average of Day before and Day after.

Figure 4.2.4: Result of Single Day Missing for Solar Farm Output Testing Set.

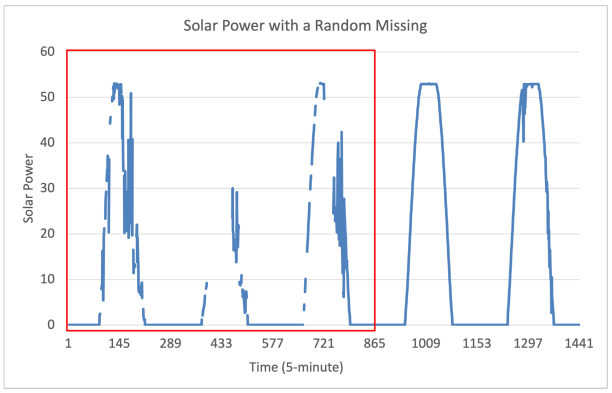


(a) Multiple Days Missing.

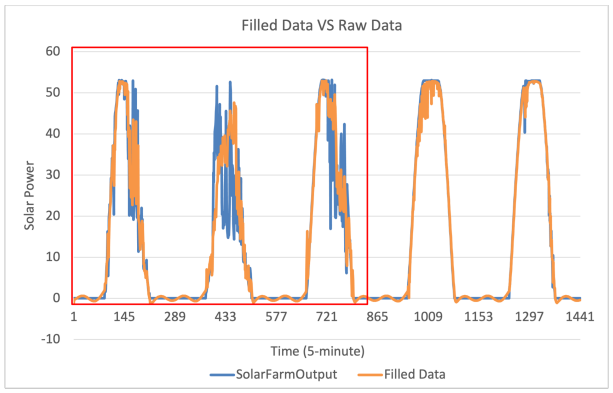


(b) Filled using Previous Year Data.

Figure 4.2.5: Result of Multiple Days Missing for Solar Farm Output Testing Set.



(a) Miscellaneous Missing Values.



(b) Fourier Model + AR(5) Model.

Figure 4.2.6: Result of Miscellaneous Missing Values for Solar Farm Output Testing Set.

Other possible variations for the second day in Figure 4.2.6 are shown in Figure 4.2.7. It could be seen that the sequences are overall similar. The variation of either of them is not large, which means there couldn't be any large fluctuations generated on the synthetic data using a normal distribution.

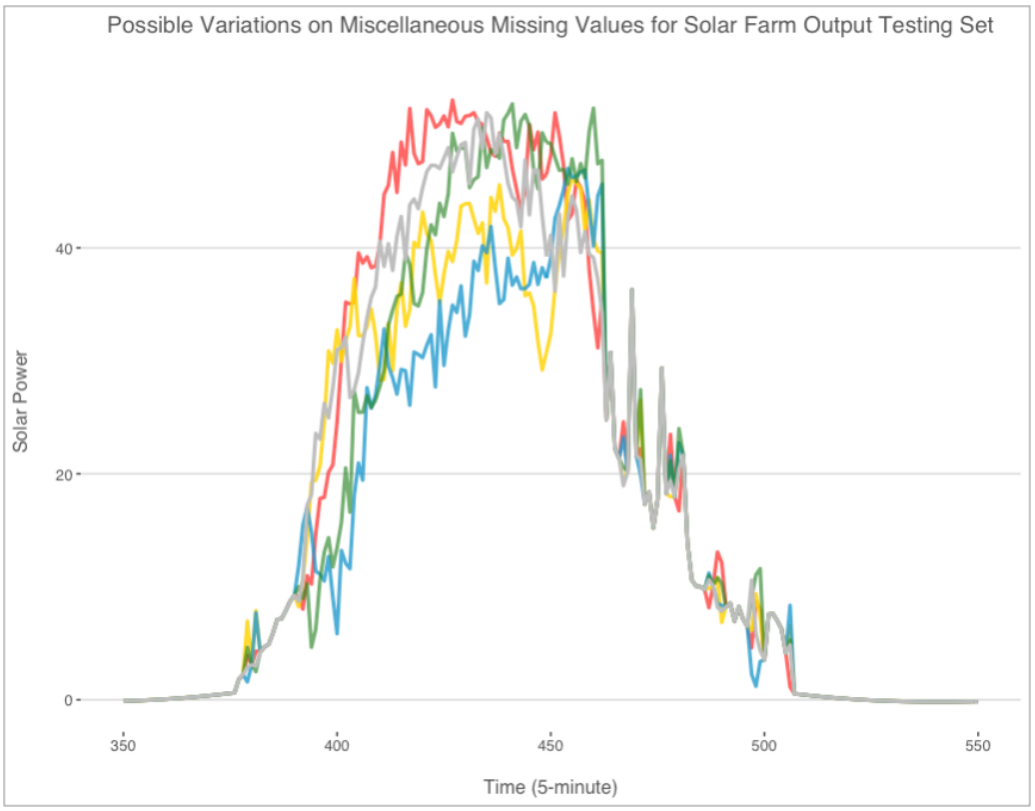


Figure 4.2.7: Possible Variations generated using Normal Distribution for Solar Farm Output Testing Set.

4.3 Infilling Results of Wind Farm Output

There is no trend or seasonality in wind farm output, so we only need to construct the AR model for this variable. In this case, 30-day data is enough to train the model.

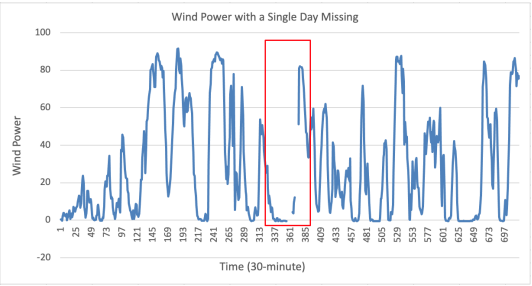
Two similar AR models are used here. The equation of the AR model for the multiple days missing is

$$C_t = 1.4378 \times C_{t-1} - 0.6042 \times C_{t-2} + 0.2482 \times C_{t-3} - 0.1096 \times C_{t-4} + \varepsilon_t.$$

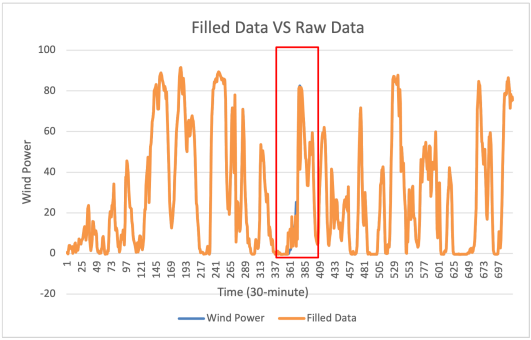
The equation of the AR model for the miscellaneous missing is

$$C_t = 1.4277 \times C_{t-1} - 0.5894 \times C_{t-2} + 0.2567 \times C_{t-3} - 0.1207 \times C_{t-4} + \varepsilon_t.$$

Figure 4.3.1, Figure 4.3.2 and Figure 4.3.3 demonstrate the results we got for wind farm output. It could be found that our filled data match the raw data for all three cases very well, even for the case when we need to simulate the synthetic data.

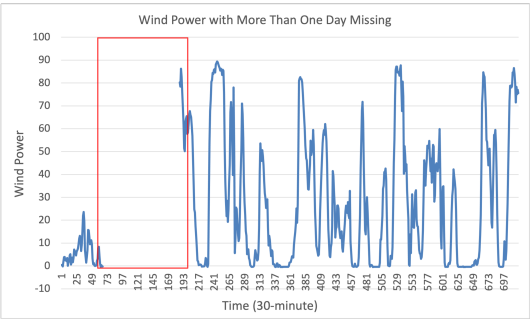


(a) Single Day Missing.

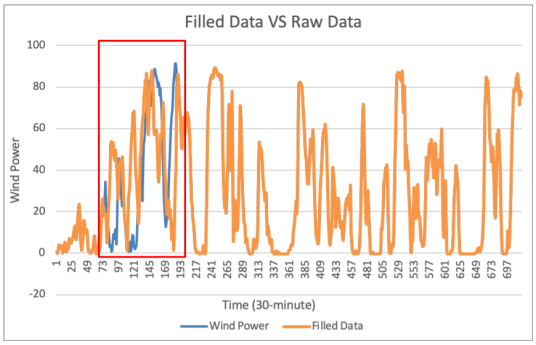


(b) Filled using Average of Day before and Day after.

Figure 4.3.1: Result of Single Day Missing for Wind Farm Output Testing Set.

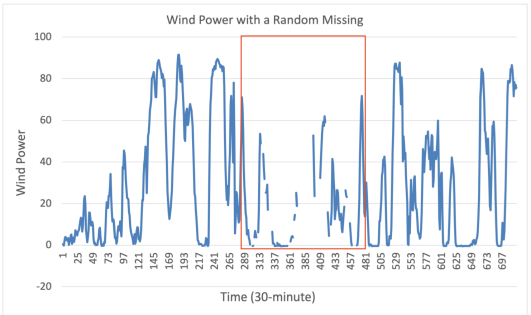


(a) Multiple Days Missing.

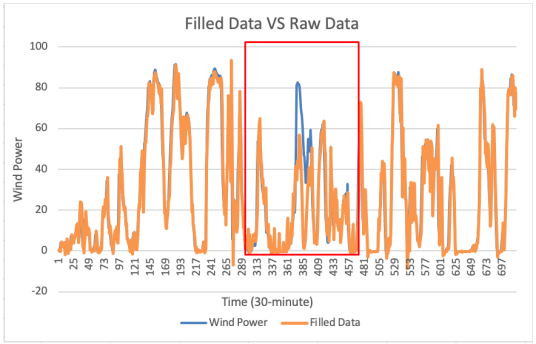


(b) Filled using AR(4) Model.

Figure 4.3.2: Result of Multiple Days Missing for Wind Farm Output Testing Set.



(a) Miscellaneous Missing Values.



(b) Filled using AR(4) Model.

Figure 4.3.3: Result of Miscellaneous Missing Values for Wind Farm Output Testing Set.

5 Conclusion

In this project, we used the additive model for solar farm output, wind farm output and ambient temperature. The strategies we used were modelling and simulating the different part of the time series based on the data characteristics. What we mean by the latter is, whether the variable has inherent trend and seasonality, or with some random noise. Then we could model the trend component using the linear model, model the seasonal component using Fourier series model, model and simulate the remaining stochastic variation using AR model. Once the models have been developed, the model results could be added to infill the missing data.

This approach is simple and flexible. There is no complicated algorithm involved, and it does not require any other year of data, sheet of data or data from neighbouring stations. We would only need one model if the variable has one component. As long as there is enough complete continuous piece of data in the data set for us to model the data characteristics of the variable, then we are able to fill the missing values of the same variable at whatever places in the data set.

However, the way we did is the simplest approach, for example, the distribution used to simulate the synthetic sequences is the most common and simplest one, the normal distribution. Thus the following work has been considered for further step

- Experiment with other distributions for the synthetic sequences, e.g., beta distribution [Grantham et al. 2018].
- Investigate the infilling methods for the multiplicative time series such as rainfall.

6 References

[Boland 2010] Boland, J 2010, 'Fourier transform', *UniSA*, https://lo.unisa.edu.au/pluginfile.php/1155213/mod_resource/content/1/Fourier%20Transform.pdf

[Boland 2020] Boland, J 2020, 'Characterising Seasonality of Solar Radiation and Solar Farm Output *Energies (Basel)*, vol. 13, no. 2, p. 471.

[Farah & Boland 2021] Farah, S & Boland, J 2021, 'Time series model for real-time forecasting of Australian photovoltaic solar farms power output', *Journal of Renewable and Sustainable Energy*, vol. 13, no. 4, p. 46102.

[Grantham et al. 2018] Grantham, A, Pudney, P & Boland, J 2018, 'Generating synthetic sequences of global horizontal irradiation *Solar Energy*, vol. 162, pp. 500–509.

[Hyndman & Athanasopoulos 2021] Hyndman, RJ & Athanasopoulos, G 2021, *Forecasting: principles and practice*, 3rd edition, OTexts: Melbourne, Australia. OTexts.com/fpp3.

[Moritz et al. 2015] Moritz, S, Sardá, A, Bartz-Beielstein, T, Zaefferer, M & Stork, J 2015, 'Comparison of different methods for univariate time series imputation in R', *ArXiv e-print*.

[Pratama et al. 2016] Pratama, I, Permanasari, AE, Ardiyanto, I & Indrayani, R 2016, 'A review of missing values handling methods on time-series data', *2016 International Conference on Information Technology Systems and Innovation (ICITSI)*, IEEE, pp. 1–6.

[Ramos-Calzado et al. 2008] Ramos-Calzado, P, Gómez-Camacho, J, Pérez-Bernal, F & Pita-López, MF 2008, 'A novel approach to precipitation series completion in climatological datasets: application to Andalusia', *International Journal of Climatology*, vol. 28, no. 11, pp. 1525–1534.

[Rubin 1976] Rubin, DB, 1976, 'Inference and missing data', *Biometrika*, vol. 63, no. 3, pp. 581–592.

[Wang 2017] Wang, JJ 2017, 'Handle Missing Values in Time Series For Beginners', *Kaggle*, https://www.kaggle.com/juejuewang/handle-missing-values-in-time-series-for-beginners

7 Appendix

7.1 Infilling Results of Ambient Temperature

Temperature is similar to solar farm output, it has no trend but seasonality. We do the similar procedure with temperature as what we did for solar farm output, and this time we built the model based on the data on 5 years, and test the model using data on another 5 years. Once-a-year frequency is quite reasonable as the summer is different from the winter.

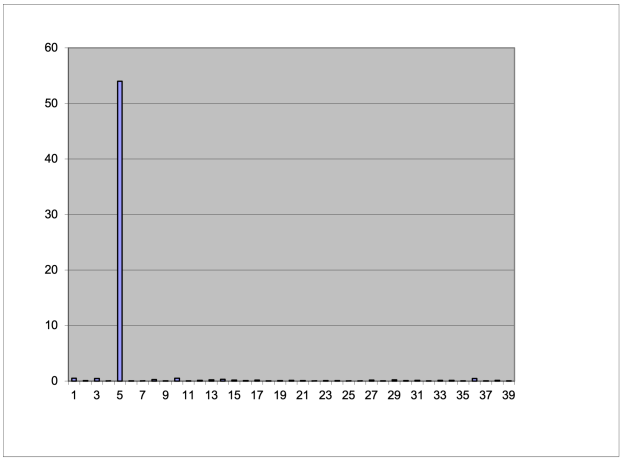


Figure 7.1.1: Ambient Temperature Power Spectrum.

From Figure 7.1.1, it could be seen that there is only one significant frequency in this 5-year series, and it stands for once a year.

The Fourier series representation of the Temperature is shown as follows:

$$F(t) = 22.98 + (a_5 cos(5 \times 2\pi t/1833) + b_5 sin(5 \times 2\pi t/1833)) \,. \tag{7.1}$$

The value of a_5 and b_5 here is 7.0969 and 2.0225.

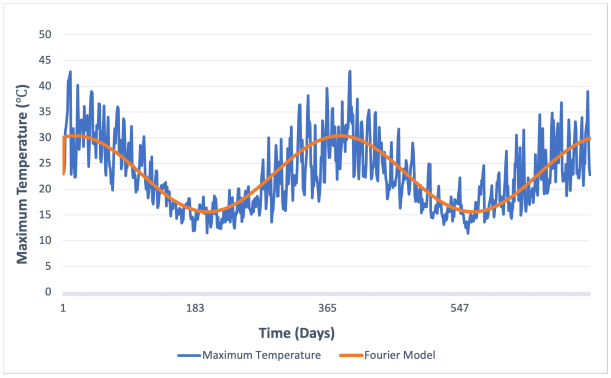
Figure 7.1.2 shows the result of Fourier series model and Fourier series model + AR(3) model applied on training set we got from Equation 7.1. It is noticeable that our approach successfully modelled the training set of temperature variable. Figure 7.1.3 is the result we got when we apply the same Fourier series model to the testing set.

Similarly, we use the average of a year before and a year after the missing values to fill a single year missing. The result is shown in Figure 7.1.4, our filled data still perfectly covers the raw data as before.

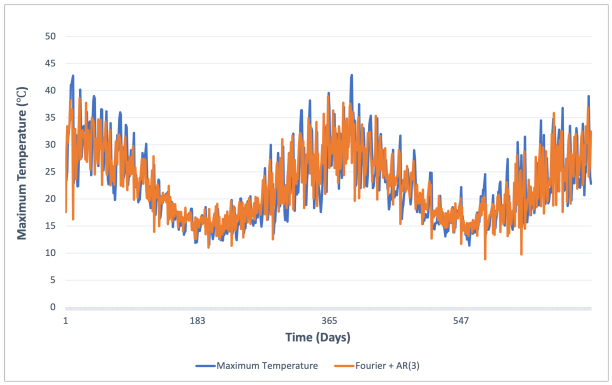
To fill the missing gaps which is shown in Figure 7.1.5, we use the synthetic generation. The AR equation here is

$$C_t = 0.6662 \times C_{t-1} - 0.1864 \times C_{t-2} + \varepsilon_t.$$

It could be seen that the stochastic variation is lying in between the minimum and maximum, however our model is not able to catch the peak of each year.



(a) Fourier Series Model.



(b) Fourier Series Model + AR(3) Model.

Figure 7.1.2: Model Results of Ambient Temperature Training Set.

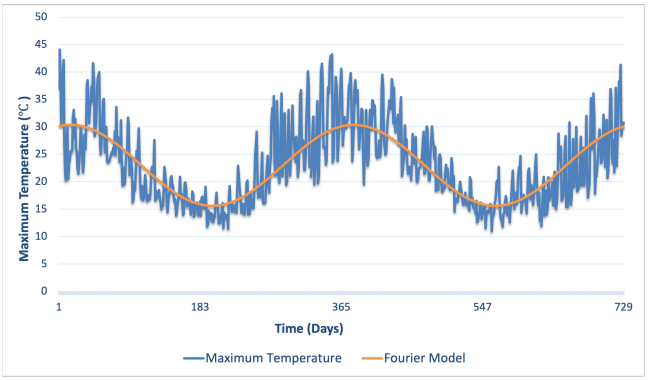
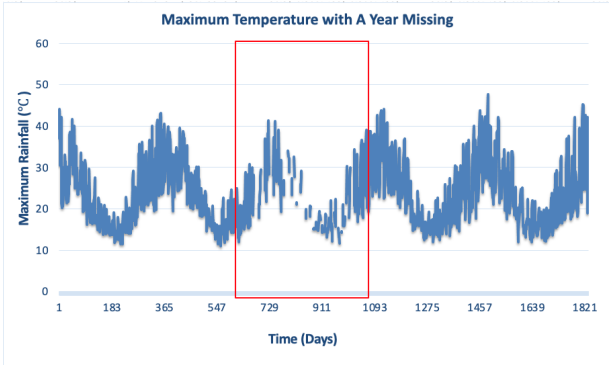
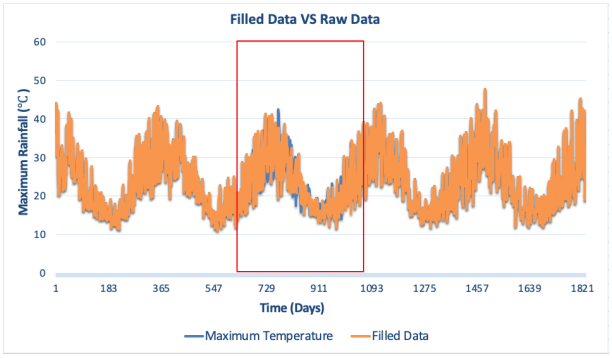


Figure 7.1.3: Fourier Series Model for Ambient Temperature Testing Set.

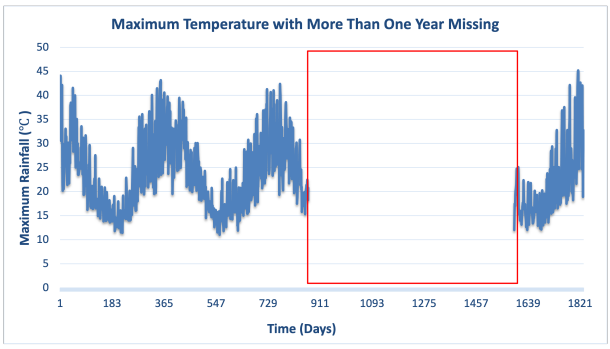


(a) Single Year Missing.

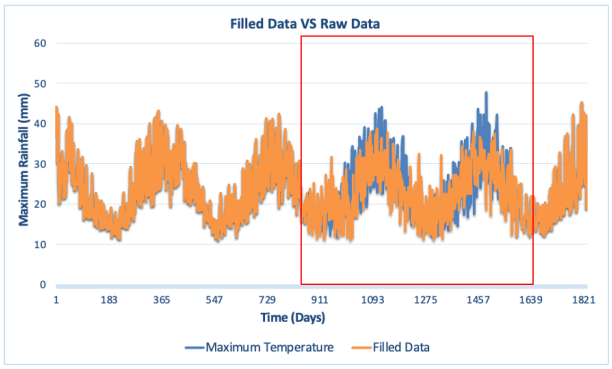


(b) Filled using Average of Year before and Year after.

Figure 7.1.4: Result of Single Year Missing for Ambient Temperature Testing Set.



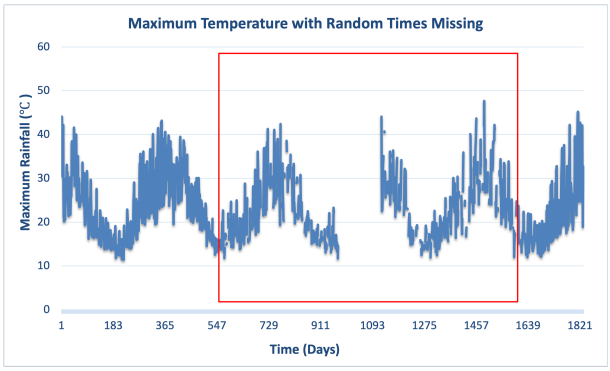
(a) Multiple Years Missing.



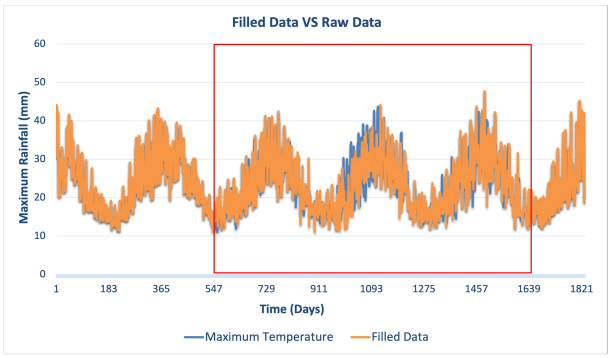
(b) Fourier Model + AR(2) Model.

Figure 7.1.5: Result of Multiple Years Missing for Ambient Temperature Testing Set.

As we find there is not much difference between the AR models of two missing patterns when we work on the wind farm output, hence we simply apply the AR model which we got from gap missing of temperature on miscellaneous missing this time. The result could be considered as an appropriate output (see Figure 7.1.6).



(a) Miscellaneous Missing Values.



(b) Fourier Model + AR(2) Model.

Figure 7.1.6: Result of Miscellaneous Missing Values for Ambient Temperature Testing Set.

7.2 An Example of R Code

The following example is the code for solar farm output, implemented in R.

```r
# AMSI 2021-2022 Winter Vacation Project
# Fourier Analysis Model and ARIMA Model for Solar Farm Output data at Broken hill.
# Author: Wang, Hanyi
#
# This code file perform the infilling approach for solar farm output
#
# Input: Data sets of solar farm output
#
# output: 1. series with newly generated gaps
#         2. series with completed gaps

# Libraries
library(readxl)
library(ggplot2)
library(dplyr)
library(tidyr)
library(writexl)
library(naniar)
library(missMethods) # for MCAR function
library(forecast) # ARIMA

# Generate NA value
## a day = 288 rows
## three days = 288*3 = 864
## five days = 288*5 = 1440
## 2 hours = 288/24*2 = 12*2 = 24

# one single value

# a single day missing
# mid-day length is 128, whole day length is 288

# more than one day missing
# missing three days in five days

# mixed
# in three days, missing percentage be 30% non-zero value, with a single mid-day and two-hour missing
generate_na_func <- function(data){

  # for reproductivity
  set.seed(2022)

  # get the row number for those non-zero values
  # we will create missing data from these non-zero values
  sunshine.row <- which(data$SolarFarmOutput != 0)

  ########### single data point##############################
  # the place for a single missing value
  randomNum1 <- sample(sunshine.row,1)

  ########### single day ###################################
```

```
# row index of sunshine.row where the sunshine of each day start
sunshine.day <- c(1, 129, 258, 392, 525)

# randomly select index of one day here for start
randomIndex2 <- sample(sunshine.day,1)
# randomly select the missing start at that day
# randomNum2.1 <- floor(runif(1, min=randomNum2, max=randomNum2+64))
# to the end of the sunshine (index as well)
randomIndex2.1 <- randomIndex2 + 127
# Make a vector for those random numbers
randomVector2 <- sunshine.row[c(randomIndex2:randomIndex2.1)]

########### 3-day block ###################################
# for 3 days missing block
# randomly select one day here for start
randomNum3 <- sample(sunshine.day,1)
randomNum3.1 <- randomNum3 + 863

########### Mixed ###################################
# row index of sunshine.row where the sunshine of each day start
sunshine.day3 <- c(1, 129, 258)
# randomly select index of one day here for start
randomIndex4 <- sample(sunshine.day3,1)
# to the end of the 3 days (index as well)
randomIndex4.1 <- randomIndex4 + (127*3)
# Make a vector for those random numbers
randomVector4 <- sunshine.row[c(randomIndex4:randomIndex4.1)]

# get a random number for this two-hour gap start
randomNum5 <- sample(randomVector4,1)
# find the corresponding index in vector
randomIndex5 <- which(randomVector4 == randomNum5)
# find the index of the end of the two weeks in vector
randomIndex5.1 <- randomIndex5 + 24
# make a vector for those randomly generated two-week rows
twoWeeksVector <-  randomVector4[c(randomIndex5:randomIndex5.1)]

# get a random number for this one-mid-day gap start
randomNum6 <- sample(sunshine.day,1)
# to the end of that mid-day
randomNum6.1 <- randomNum6 + 70

########### generator ###################################
data_missing <- data %>%
  mutate(single = SolarFarmOutput) %>%
  mutate(oneDay = SolarFarmOutput) %>%
  mutate(threeDays = SolarFarmOutput) %>%
  mutate(mixed = SolarFarmOutput)

# randomly create missing values in the mid-day
data_missing[randomVector2,"oneDay"] <- delete_MCAR(data_missing[randomVector2,"SolarFarmOutput"], 0.5, "
  SolarFarmOutput")
data_missing[randomVector4,"mixed"] <- delete_MCAR(data_missing[randomVector4,"SolarFarmOutput"], 0.3, "
  SolarFarmOutput")

```

```
  # generate na values
  data_missing$single[randomNum1] <- NA
  data_missing$threeDays[c(randomNum3:randomNum3.1)] <- NA
  data_missing$mixed[twoWeeksVector] <- NA
  data_missing$mixed[c(randomNum6:randomNum6.1)] <- NA

  return(data_missing)
}

################################################################################
################################################################################
# Read Solar Farm Data
solarfarm.year2 <- read_excel("SolarFarm.xlsx", sheet = 3)
colnames(solarfarm.year2) <- "SolarFarmOutput"

# cut the data, only keep first five days
solarfarm.year2.2 <- solarfarm.year2[c(1:1441),]

# make the data frame for the missing records
solarfarm.year2.missing <- generate_na_func(solarfarm.year2.2)

# check the na value - correct
df <- solarfarm.year2.missing[,2]
df <- df[rowSums(is.na(df)) > 0,]

# check mixed missing data
plot(ts(solarfarm.year2.missing$mixed))

write_xlsx(solarfarm.year2.missing,"~/Desktop/TestFiveDaysWithMissing.xlsx")

################################################################################
################################################################################
# Cycles
cycle4 <- 5*2*pi/1440
cycle5 <- 10*2*pi/1440
cycle6 <- 20*2*pi/1440

# 5 cycles
par4.1 <- -22.7960057563174
par4.2 <- -4.25335259753835

# 10 cycles
par5.1 <- 9.43694889034436
par5.2 <- 3.87286813681687

# 20 cycles
par6.1 <- -4.01370195966213
par6.2 <- -2.43247508659813

# average
avg <- 14.50461

# create seasonalities
solarfarm.year2.model <- solarfarm.year2.missing %>%
  mutate(ID = row_number()) %>% # generate id columns
```

```
  mutate(s4 = par4.1 * cos(cycle4*ID) + par4.2 * sin(cycle4*ID)) %>%
  mutate(s5 = par5.1 * cos(cycle5*ID) + par5.2 * sin(cycle5*ID)) %>%
  mutate(s6 = par6.1 * cos(cycle6*ID) + par6.2 * sin(cycle6*ID)) %>%
  mutate(model = avg + s4 + s5 + s6) %>%
  mutate(r1 = mixed - model)

# check the Fourier Model
plot(ts(solarfarm.year2.model$model))

##############################################################################
##############################################################################
# AR Model from previous year
solarfarm.pre <- read_excel("SolarFarm.xlsx", sheet = 2)

# col12 = cycle 5 + cycle 10 + cycle 20
solarfarm.pre.r1 <- as.numeric(pull(solarfarm.pre[c(3:1443),12]))
auto.arima(solarfarm.pre.r1) # 5,0,3 - AIC 8060
# col 12: AR(5)-8031 AR(4)- 8046 AR(3)-AIC 8044 AR(2) - 8047
arima.fit.pre <- Arima(solarfarm.pre.r1, order=c(5,0,0))

# coeficients
unlist(arima.fit.pre['coef'])

# model 2 fitted value
model2 <- as.data.frame(unlist(arima.fit.pre['fitted']))
write_xlsx(model2,"~/Desktop/model2.xlsx")

##############################################################################
##############################################################################
# AR model - for mixed gaps
auto.arima(solarfarm.year2.model$r1) # AIC 6179
#AR(3): 6067 AR(2): 6137 1:6233 4:6046 5:6030
arima.fit <- Arima(solarfarm.year2.model$r1, order=c(5,0,0), include.constant=FALSE)

# AR(5)
ar.pars <- unlist(arima.fit['coef'])
ar1 = ar.pars[1]
ar2 = ar.pars[2]
ar3 = ar.pars[3]
ar4 = ar.pars[4]
ar5 = ar.pars[5]

# add the Ar(5) model to the df
solarfarm.year2.model$model2 <- unlist(arima.fit['fitted'])
summary(solarfarm.year2.model$model2 - solarfarm.year2.model$r1)

# make a copy for data frame
model.copy <- solarfarm.year2.model

# maximum solar farm output
M <- max(na.omit(solarfarm.year2.missing$mixed))

# residuals
residuals <- unlist(arima.fit["residuals"])
hist(residuals)
```

```r
# mean and var of this error terms with ignoring nas
residuals.var <- var(na.omit(residuals))
residuals.mean <- mean(na.omit(residuals))

set.seed(2022)
for (row in 1:nrow(model.copy)) {
  if(is.na(model.copy$model2[row]) == TRUE){

    # if lag1 and lag2 are existed
    if(is.na(model.copy$model2[(row-1)]) == FALSE & is.na(model.copy$model2[(row-2)]) == FALSE){

      model.copy$model2[row] = ar1*as.vector(model.copy$model2[(row-1)]) +
        ar2*as.vector(model.copy$model2[(row-2)]) +
        ar3*as.vector(model.copy$model2[(row-3)]) +
        ar4*as.vector(model.copy$model2[(row-4)]) +
        ar5*as.vector(model.copy$model2[(row-5)])

      # Add random noise
      model.copy$model2[row] =  model.copy$model2[row] + rnorm(1, mean = residuals.mean, sd = sqrt(residuals.var))

      # Add constraint on filled data
      while (model.copy$model2[row] + model.copy$model[row] > M | model.copy$model2[row] + model.copy$model[row] < 0) {
        model.copy$model2[row] = ar1*as.vector(model.copy$model2[(row-1)]) +
          ar2*as.vector(model.copy$model2[(row-2)]) +
          ar3*as.vector(model.copy$model2[(row-3)]) +
          ar4*as.vector(model.copy$model2[(row-4)]) +
          ar5*as.vector(model.copy$model2[(row-5)]) + rnorm(1, mean = residuals.mean, sd = sqrt(residuals.var))
      }
    }
  }
}

# additive model
model.copy$finalModel3 = model.copy$model + model.copy$model2

# check the fitted
# model fit
summary(model.copy$SolarFarmOutput - model.copy$finalModel3)
# check the plot
plot(1:1441,                              # Draw first time series
     model.copy$SolarFarmOutput,
     type = "l",
     col = 2,
     xlab = "Time",
     ylab = "SolarFarmOutput")
lines(1:1441,                             # Draw second time series
      model.copy$model,
      type = "l",
      col = 3)
lines(1:1441,                             # Draw third time series
      model.copy$finalModel3,
      type = "l",
```

```
      col = 4)
legend("bottomright",                          # Add legend to plot
       c("Raw data", "Fourier Model", "Fourier Model + AR(2)"),
       lty = 1,
       col = 2:4)

# plot
plot(1:900,                                # Draw first time series
     model.copy$SolarFarmOutput[1:900],
     type = "l",
     col = "dodgerblue3",
     lwd=2,
     xlab = "Time",
     ylab = "SolarFarmOutput")
lines(1:900,                               # Draw third time series
      model.copy$finalModel3[1:900],
      lwd=2,
      type = "l",
      col = "chocolate1")
legend("bottomleft",                           # Add legend to plot
       c("Raw data","Fourier Model + AR(2)"),
       lty = 1,
       col = c("dodgerblue3","chocolate1"))

################################################################################
################################################################################
# for single one day gap
# mean between row-288 and row+288

# make a copy for the data frame and target column
model.copy.single <- model.copy %>%
  mutate(modelSingleDay = oneDay)

# define the day length
day.length <- 288

# find the na in a loop for the row as before
for (row in 1:nrow(model.copy.single)) {
  if(is.na(model.copy.single$modelSingleDay[row]) == TRUE){
    model.copy.single$modelSingleDay[row] = mean(c(model.copy.single$modelSingleDay[row-287], model.copy.
     single$modelSingleDay[row+287]))

  }
}
################################################################################
################################################################################
# for more than one day gap
# copy the days from previous year and paste to this year
# rows <- c(258:1122)

# copy the days from previous year
rows.pre <- as.numeric(pull(solarfarm.pre[258+2:1122+2,2]))
# make a copy for thr data frame
model.copy.final <- model.copy.single %>%
  mutate(modelThreeDays = threeDays)
```

```
321
322 # paste to this year
323 model.copy.final$modelThreeDays[258:1122] <- rows.pre
324
325 # plot
326 plot(1:1441,                              # Draw first time series
327      model.copy.final$SolarFarmOutput[1:1441],
328      type = "l",
329      col = "dodgerblue3",
330      lwd=2,
331      xlab = "Time",
332      ylab = "SolarFarmOutput")
333 lines(1:1441,                             # Draw third time series
334       model.copy.final$modelThreeDays[1:1441],
335       lwd=2,
336       type = "l",
337       col = "chocolate1")
338 legend("bottomleft",                            # Add legend to plot
339        c("Raw data","Filled Data"),
340        lty = 1,
341        col = c("dodgerblue3","chocolate1"))
342
343 ##############################################################################
344 ##############################################################################
345 write_xlsx(model.copy,"~/Desktop/final1.xlsx")
346 write_xlsx(model.copy.single,"~/Desktop/final3.xlsx")
```