Document for *Historical Reconstruction Dataset of Hourly Expected On-Shore Wind Generation in Japan*

Introduction

- This is a supplemental document to the files published at zenodo.
- The dataset consists of a set of netCDF files with yearly archives of reconstruction results from 1958 to 2012; hourly expected on-shore wind power potential in Japan with a spatial resolution of approximately 5 km mesh has been reconstructed from the numerical weather model reanalysis results. The expected per-unit output values at each location were calibrated using a nonparametric machine learning model that learns statistical relationships between spatial/meteorological features of target locations and actual wind farm outputs.
- A convenient way to handle this dataset would be to use a tool for manipulating netCDF files, such as CDO: Climate Data Operators.
- This document describes other ways to extract information from this dataset in Python or R languages without using the above tool.

Version history

Ver. 1.0:

- Released on August 4, 2023.

Ver. 1.1:

- Released on October 31, 2023.
  - The preprocessing of the source information used for dataset preparation has changed.

Ver. 1.2:

- Released on June 7, 2024.
  - The hyperparameter tuning scheme for the post-processing model has changed.

Dataset information

- Fujimoto, Y., Ohba, M., Kanno, Y., Nohara, D. & Y. Hayashi (2024). Historical Reconstruction Dataset of Hourly Expected On-Shore Wind Generation in Japan (Ver. 1.2) [Data set]. Zenodo.
- Please see https://doi.org/10.5281/zenodo.11496867 for further information.

Extract netCDF file from archive

- Firstly, we recommend to extract netCDF file from the archive (e.g., `windpower_1959_1hr.tar.gz`) for handling.

```shell
$ tar xvzf windpower_1959_1hr.tar.gz
```

Sample snippets for Python

Preparation

- To handle netCDF file on Python, we need `netCDF4` https://unidata.github.io/netcdf4-python/
- To install `netCDF4` package via `pip` command, we can use the following:

SHELL

```
$ pip install netCDF4
```

Check basic information

- The following Python snippets have been tested by using Python 3.10.12.

Check data structure

- An easy way to read a netCDF file is to use `netCDF4.Dataset()` function.

PYTHON

```
import netCDF4

trgfile = 'windpower_1959_1hr.nc'
nc = netCDF4.Dataset(trgfile,'r')
print(nc)
```

- The file 'windpower_1959_1hr.nc' includes the 519×419 meshed information of the hourly expected wind power in per-unit for $365 \times 24 = 8760$ timeslices.
- We can see the following data structure

```
<class 'netCDF4._netCDF4.Dataset'>
root group (NETCDF4 data model, file format HDF5):
    dimensions(sizes): lon(519), lat(419), time(8760)
    variables(dimensions): float64 lon(lon), float64 lat(lat), float64 time(time),
float32 windpower(time, lat, lon)
    groups:
```

Time component

- Time slice information is included in the 'time' component.

PYTHON

```
time_component = nc.variables['time']
print(time_component)
```

- We can see the specification of the 'time' component in the target netCDF file.

```
<class 'netCDF4._netCDF4.Variable'>
float64 time(time)
    long_name: Time
    units: minutes since 1959-01-01 00:00
unlimited dimensions: time
current shape = (8760,)
filling on, default _FillValue of 9.969209968386869e+36 used
```

- We can access the elements in the 'time' component as the (masked_)array form.
  - For example, `time_component[:][0:3]` returns the following:

```
masked_array(data=[  0.,  60., 120.],
             mask=False,
       fill_value=1e+20)
```

- The elements in the 'time' component indicates the minutes since yyyy-mm-dd 00:00 (UTC).

Coordinate information

- In the same manner, we can access the mesh coordinate information as follows.

PYTHON

```python
longitude = nc.variables['lon']
latitude = nc.variables['lat']
```

Expected wind power information

- The 'windpower' component returns the target expected wind power information.

PYTHON

```python
w = nc.variables['windpower']
```

- `w.shape` returns its dimension as follows:

```
(8760, 419, 519)
```

Some sample Python snippets

Ex. 1) Extract the annual sequence of the specific point

- We focus on the specific target point (longitude:139.7217, latitude:35.7103) for example.
- To extract the annual sequence of the expected wind power in 1959 at the nearest mesh point to the target point as the csv file, we can use the following Python snippet.

PYTHON

```python
import numpy as np
import pandas as pd
import netCDF4

trgfile = 'windpower_1959_1hr.nc'
trg_lon = 139.7217
trg_lat = 35.7103

nc = netCDF4.Dataset(trgfile,'r')
lon_id = np.argmin(np.abs(nc.variables['lon'][:]-trg_lon))
lat_id = np.argmin(np.abs(nc.variables['lat'][:]-trg_lat))
w = nc.variables['windpower'][:,lat_id,lon_id]
df = pd.DataFrame(w)
df.to_csv('output.csv',index=False)
```

Ex. 2) Extract the snapshot representing spatial distribution of the specific timing

- We focus on the first timeslice included in 'windpower_1959_1hr.nc' file (i.e., time:1959-01-01 00:00) for example.
- To extract the snapshot as the csv file, we can use the following snippet.

PYTHON

```python
import pandas as pd
import netCDF4

trgfile = 'windpower_1959_1hr.nc'
timeslice = 0 # the first timeslice

nc = netCDF4.Dataset(trgfile,'r')
w = nc.variables['windpower'][timeslice,:,:]
df=pd.DataFrame(w)
df.to_csv('snapshot.csv',index=False)
```

Sample snippets for R

Preparation

- To handle netCDF file on R, we can use `ncdf4` https://cran.r-project.org/web/packages/ncdf4/index.html or `RNetCDF` https://cran.r-project.org/web/packages/RNetCDF/index.html .
- In this document, we use `RNetCDF`.
- To install `RNetCDF` package, we can use the following:

R

```r
install.packages("RNetCDF")
```

Check basic information

- The following R snippets have been tested by using R 4.2.0.

Check data structure

- An easy way to read a netCDF file is to use `open.nc()` function (`close.nc()` will be also required at the end of operation.)

R

```r
library(RNetCDF)

trgfile <- 'windpower_1959_1hr.nc'
nc <- open.nc(trgfile)
print.nc(nc)
```

- The file "windpower_1959_1hr.nc" includes the 519×419 meshed information of the hourly expected wind power in per-unit for $365 \times 24 = 8760$ timeslices.
- We can see the following information by `print.nc(nc)`

```
netcdf netcdf4 {
dimensions:
        lon = 519 ;
        lat = 419 ;
        time = UNLIMITED ; // (8760 currently)
variables:
        NC_DOUBLE lon(lon) ;
                NC_CHAR lon:long_name = "Longitude" ;
                NC_CHAR lon:units = "degrees_east" ;
        NC_DOUBLE lat(lat) ;
                NC_CHAR lat:long_name = "Latitude" ;
                NC_CHAR lat:units = "degrees_north" ;
        NC_DOUBLE time(time) ;
                NC_CHAR time:long_name = "Time" ;
                NC_CHAR time:units = "minutes since 1959-01-01 00:00" ;
        NC_FLOAT windpower(lon, lat, time) ;
                NC_CHAR windpower:long_name = "Wind Power" ;
                NC_CHAR windpower:units = "pu" ;
}
```

Time component

- Time slice information is included in the 'time' component.

```R
time_component <- var.get.nc(nc,"time")
head(time_component)
```

- We can see the specification of the 'time' component in the target netCDF file.

```
[1]   0  60 120 180 240 300
```

- The elements in the 'time' component indicates the minutes since yyyy-mm-dd 00:00 (UTC).

Coordinate information

- In the same manner, we can access the mesh coordinate information as follows.

```R
longitude <- var.get.nc(nc,"lon")
latitude <- var.get.nc(nc,"lat")
```

Expected wind power information

- The 'windpower' component returns the target expected wind power information.

```R
w <- var.get.nc(nc,"windpower")
```

- `dim(w)` returns its dimension as follows:

```
[1]  519  419 8760
```

Close the file connection

- Do not forget `close.nc()` at the end of operation.

```R
close.nc(nc)
```

Some sample R snippets

Ex. 1) Extract the annual sequence of the specific point

- We focus on the specific target point (longitude:139.7217, latitude:35.7103) for example.
- To extract the annual sequence of the expected wind power in 1959 at the nearest mesh point to the target point as the csv file, we can use the following R snippet.

```R
library(RNetCDF)

trgfile <- "windpower_1959_1hr.nc"
trg_lon <- 139.7217
trg_lat <- 35.7103

nc <- open.nc(trgfile)
longitude <- var.get.nc(nc,"lon")
latitude <- var.get.nc(nc,"lat")
wp <- var.get.nc(nc,"windpower")
close.nc(nc)

lon_id <- which.min(abs(longitude-trg_lon))
lat_id <- which.min(abs(latitude-trg_lat))
w <- wp[lon_id,lat_id,]
write.table(w,"output.csv",sep=",",row.names=FALSE,col.names=FALSE)
```

Ex. 2) Extract the snapshot representing spatial distribution of the specific timing

- We focus on the first time slice included in 'windpower_1959_1hr.nc' file (i.e., time:1959-01-01 00:00) for example.
- To extract the snapshot as the csv file, we can use the following snippet.

R

```r
library(RNetCDF)

trgfile <- "windpower_1959_1hr.nc"
timeslice = 1 # the first timeslice

nc <- open.nc(trgfile)
wp <- var.get.nc(nc,"windpower")
close.nc(nc)

w <- wp[,,timeslice]
write.table(w,"snapshot.csv",sep=",",row.names=FALSE,col.names=FALSE)
```