A Collection of Python Examples

Fan Wang

2020-09-01

Contents

Preface

The work-in-progress pyfan repository contains:

1. Tutorials and examples for various research tasks: **bookdown site** and **bookdown pdf**.
2. A package for basic data, graph and research tasks: **readthedocs** and **pypi**.

Materials are gathered from various projects in which python code is used for research and paper-administrative tasks. Files are from **Fan**'s pyfan repository which has an associated package. The package functionalize various tasks tested out in the Rmd files. In addition, the pyecon repository and the associated package (readthedocs) contain functions and rmd files related explicitly to solving economic models.

From Fan's other repositories: For dynamic borrowing and savings problems, see Dynamic Asset Repository (Matlab); For code examples, see also Matlab Example Code, R Example Code, and Stata Example Code; For intro econ with Matlab, see Intro Mathematics for Economists, and for intro stat with R, see Intro Statistics for Undergraduates. See here for all of Fan's public repositories.

The site is built using Bookdown (Xie, 2020).

Please contact FanWangEcon for issues or problems.

Chapter 1

Array, Matrix, Dataframe

1.1 Array

1.1.1 Strings

Go back to fan's Python Code Examples Repository (bookdown site).

1.1.1.1 Search if Names Include Strings

Given a list of strings, loop but skip if string contains elements string list.

```
# define string
ls_st_ignore = ['abc', 'efg', 'xyz']
ls_st_loop = ['ab cefg sdf', '12345', 'xyz', 'abc xyz', 'good morning']

# zip and loop and replace
for st_loop in ls_st_loop:
  if sum([st_ignore in st_loop for st_ignore in ls_st_ignore]):
    print('skip:', st_loop)
  else:
    print('not skip:', st_loop)
```

```
## skip: ab cefg sdf
## not skip: 12345
## skip: xyz
## skip: abc xyz
## not skip: good morning
```

1.1.1.2 Replace a Set of Strings in String

Replace terms in string

```
# define string
st_full = """
abc is a great efg, probably xyz. Yes, xyz is great, like efg.
eft good, EFG capitalized, efg good again.
A B C or abc or ABC. Interesting xyz.
"""

# define new and old
ls_st_old = ['abc', 'efg', 'xyz']
ls_st_new = ['123', '456', '789']

# zip and loop and replace
for old, new in zip(ls_st_old, ls_st_new):
```

```
  st_full = st_full.replace(old, new)

# print
print(st_full)
```

```
##
## 123 is a great 456, probably 789. Yes, 789 is great, like 456.
## eft good, EFG capitalized, 456 good again.
## A B C or 123 or ABC. Interesting 789.
```

1.1.1.3 Wrap String with Fixed Width

Given a long string, wrap it into multiple lines with fixed width.

```
import textwrap

# A long Path
st_path = """
C:/Users/fan/Documents/Dropbox (UH-ECON)/Project Emily Minority Survey/EthLang/reg_lang_abi_cls_mino
"""

# Wrap text with tight width
st_wrapped = textwrap.fill(st_path, width = 20)
print(st_wrapped)
```

```
##  C:/Users/fan/Docume
## nts/Dropbox (UH-
## ECON)/Project Emily
## Minority Survey/EthL
## ang/reg_lang_abi_cls
## _mino/tab3_fm/attain
## _m_vs_f/tab3_mand_ta
## lk_m2c_hfracle02.tex
```

Combine Strings that are wrapped and not Wrapped

```
# Paths
st_path_a = "C:/Users/fan/Documents/Dropbox (UH-ECON)/Project Emily Minority Survey/EthLang/reg_lang
st_path_b = 'C:/Users/fan/R4Econ/support/development/fs_packaging.html'

# Combine Strings and Wrap
str_dc_records = 'First Path:'.upper() + '\n' + \
                 textwrap.fill(st_path_a, width=25) + '\n\n' + \
                 'Second Path:'.upper() + '\n' + \
                 textwrap.fill(st_path_b, width=25)

# Print
print(str_dc_records)
```

```
## FIRST PATH:
## C:/Users/fan/Documents/Dr
## opbox (UH-ECON)/Project
## Emily Minority Survey/Eth
## Lang/reg_lang_abi_cls_min
## o/tab3_fm/attain_m_vs_f/t
## ab3_mand_talk_m2c_hfracle
## 02.tex
##
## SECOND PATH:
## C:/Users/fan/R4Econ/suppo
```

```
## rt/development/fs_packagi
## ng.html
```

1.2 Dictionary

1.2.1 Dictionary

Go back to fan's Python Code Examples Repository (bookdown site).

1.2.1.1 Create a List of Dictionaries

```
import datetime
import pprint
ls_dc_exa =  [
    {"file": "mat_matlab",
     "title": "One Variable Graphs and Tables",
     "description": "Frequency table, bar chart and histogram",
     "val": 1,
     "date": datetime.date(2020, 5, 2)},
    {"file": "mat_two",
     "title": "Second file",
     "description": "Second file.",
     "val": [1, 2, 3],
     "date": datetime.date(2020, 5, 2)},
    {"file": "mat_algebra_rules",
     "title": "Opening a Dataset",
     "description": "Opening a Dataset.",
     "val": 1.1,
     "date": datetime.date(2018, 12, 1)}
]
pprint.pprint(ls_dc_exa, width=1)
```

```
## [{'date': datetime.date(2020, 5, 2),
##   'description': 'Frequency '
##                  'table, '
##                  'bar '
##                  'chart '
##                  'and '
##                  'histogram',
##   'file': 'mat_matlab',
##   'title': 'One '
##            'Variable '
##            'Graphs '
##            'and '
##            'Tables',
##   'val': 1},
##  {'date': datetime.date(2020, 5, 2),
##   'description': 'Second '
##                  'file.',
##   'file': 'mat_two',
##   'title': 'Second '
##            'file',
##   'val': [1,
##           2,
##           3]},
##  {'date': datetime.date(2018, 12, 1),
##   'description': 'Opening '
##                  'a '
```

```
##                 'Dataset.',
##   'file': 'mat_algebra_rules',
##   'title': 'Opening '
##            'a '
##            'Dataset',
##   'val': 1.1}]
```

1.2.1.2 Iteratively Add to A Dictionary

Iteratively add additional Key and Value pairs to a dictionary.

```
ls_snm_tex = ["file1.tex", "file2.tex", "file3.tex"]
ls_snm_pdf = ["file1.pdf", "file2.pdf", "file3.pdf"]

dc_tex_pdf = {}
for tex, pdf in zip(ls_snm_tex, ls_snm_pdf):
  dc_tex_pdf[tex] = pdf

pprint.pprint(dc_tex_pdf, width=1)
```

```
## {'file1.tex': 'file1.pdf',
##  'file2.tex': 'file2.pdf',
##  'file3.tex': 'file3.pdf'}
```

1.2.1.3 Select by Keys in Dictionary

Given a list of dictionary, search if key name is in list:

```
# string to search through
ls_str_file_ids = ['mat_matlab', 'mat_algebra_rules']
# select subset
ls_dc_selected = [dc_exa
                  for dc_exa in ls_dc_exa
                  if dc_exa['file'] in ls_str_file_ids]
# print
pprint.pprint(ls_dc_selected, width=1)
```

```
## [{'date': datetime.date(2020, 5, 2),
##   'description': 'Frequency '
##                  'table, '
##                  'bar '
##                  'chart '
##                  'and '
##                  'histogram',
##   'file': 'mat_matlab',
##   'title': 'One '
##            'Variable '
##            'Graphs '
##            'and '
##            'Tables',
##   'val': 1},
##  {'date': datetime.date(2018, 12, 1),
##   'description': 'Opening '
##                  'a '
##                  'Dataset.',
##   'file': 'mat_algebra_rules',
##   'title': 'Opening '
##            'a '
##            'Dataset',
##   'val': 1.1}]
```

Search and Select by Multiple Keys in Dictionary. Using two keys below:

```
# string to search through
ls_str_file_ids = ['mat_matlab', 'mat_algebra_rules']
# select subset
ls_dc_selected = [dc_exa
                  for dc_exa in ls_dc_exa
                  if ((dc_exa['file'] in ls_str_file_ids)
                      and
                      (dc_exa['val']== 1))]
# print
pprint.pprint(ls_dc_selected, width=1)
```

```
## [{'date': datetime.date(2020, 5, 2),
##   'description': 'Frequency '
##                  'table, '
##                  'bar '
##                  'chart '
##                  'and '
##                  'histogram',
##   'file': 'mat_matlab',
##   'title': 'One '
##            'Variable '
##            'Graphs '
##            'and '
##            'Tables',
##   'val': 1}]
```

Chapter 2

Tables and Graphs

2.1 Matplotlib Base Plots

2.1.1 Line and Scatter Plots

Go back to fan's Python Code Examples Repository (bookdown site).

2.1.1.1 Plot Random Walk and White Noise Jointly

Given x and y coordinates, plot out two lines. see matplotlib.pyplot.plot. Here we will plot out the extremes of AR(1), white noise (no persistence), and random walk (fully persistent shocks).

```
# Import Packages
import numpy as np
import matplotlib.pyplot as plt

# Generate X and Y
np.random.seed(123)
ar_fl_y1_rand = np.random.normal(0, 2, 100)
ar_fl_y2_rand = np.cumsum(np.random.normal(0, 1, 100))
ar_it_x_grid = np.arange(1,len(ar_fl_y1_rand)+1)

# Start Figure
fig, ax = plt.subplots()

# Graph
ax.plot(ar_it_x_grid, ar_fl_y1_rand,
                      color='blue', linestyle='dashed',
                      label='sd=2, 0 persistence')
```

```
## [<matplotlib.lines.Line2D object at 0x00000246ECC4C370>]
```

```
ax.plot(ar_it_x_grid, ar_fl_y2_rand,
                      color='red', linestyle='solid',
                      label='sd=1, random walk')

# Labeling
```

```
## [<matplotlib.lines.Line2D object at 0x00000246EC9A61F0>]
```

```
ax.legend(loc='upper left')
```

```
## <matplotlib.legend.Legend object at 0x00000246ECC4C190>
```

```
plt.ylabel('Random Standard Normal Draws')
```

```
## Text(0, 0.5, 'Random Standard Normal Draws')
```

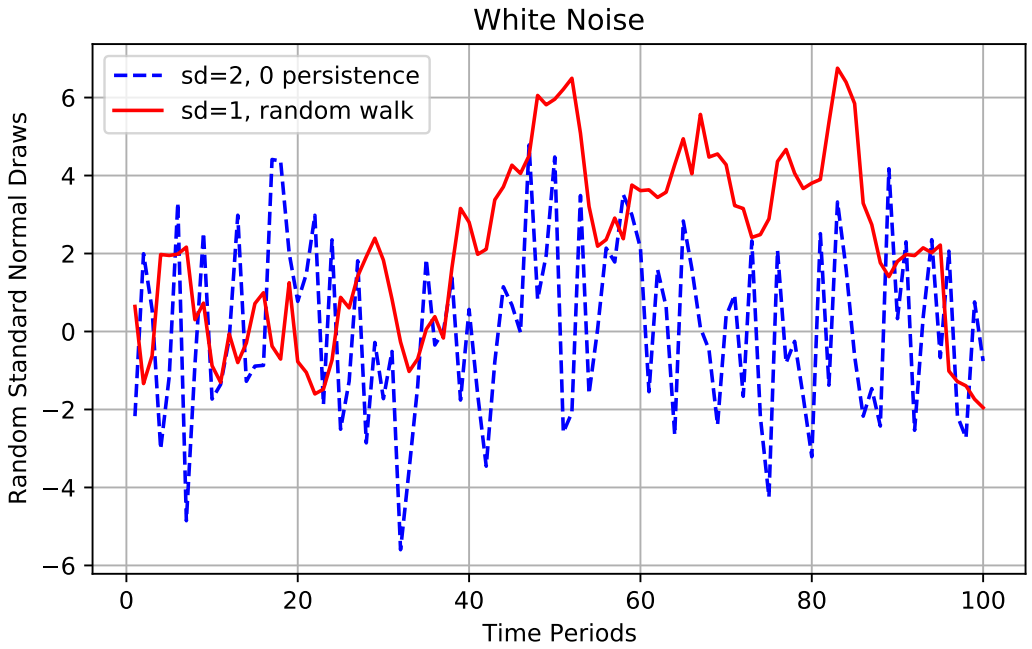
```
plt.xlabel('Time Periods')
```

```
## Text(0.5, 0, 'Time Periods')
```

```
plt.title('White Noise')
```

```
## Text(0.5, 1.0, 'White Noise')
```

```
plt.grid()
plt.show()
```



2.1.2 Text Plot

Go back to fan's Python Code Examples Repository (bookdown site).

2.1.2.1 Plot Text

Plot Text as Image. Create text with different alignment and rotation.

```
# Import Packages
import matplotlib.pyplot as plt
import textwrap
import json

# Dict of String to String
dc_path = {'C:\\Users\\fan\\Documents\\Dropbox (UH-ECON)\\repos\\Tex4Econ\\'
           '_other\\equation\\cases.tex':
               'C:/Users/fan/Documents/cases.pdf',
           'C:\\Users\\fan\\Documents\\Dropbox (UH-ECON)\\repos\\Tex4Econ\\'
           '_other\\symbols\\fs_symbols.tex':
               'C:/Users/fan/Documents/fs_symbols.pdf'}
st_dc_path = textwrap.fill(json.dumps(dc_path), width = 70)

# Start Plot
fig, ax = plt.subplots()
```

```
# Text Plot
ax.text(0.5, 0.5, st_dc_path,
        horizontalalignment='center',
        verticalalignment='center',
        fontsize=14, color='black',
        transform=ax.transAxes)

# Labeling
```

```
## Text(0.5, 0.5, '{"C:\\\\Users\\\\fan\\\\Documents\\\\Dropbox (UH-\nECON)\\\\repos\\\\Tex4Econ\\\\
```

```
ax.set_axis_off()
plt.show()
```

{"C:\\Users\\fan\\Documents\\Dropbox (UH-
ECON)\\repos\\Tex4Econ_other\\equation\\cases.tex":
"C:/Users/fan/Documents/cases.pdf",
"C:\\Users\\fan\\Documents\\Dropbox (UH-
ECON)\\repos\\Tex4Econ_other\\symbols\\fs_symbols.tex":
"C:/Users/fan/Documents/fs_symbols.pdf"}

Chapter 3

Get Data

3.1 Environmental Data

3.1.1 ECMWF ERA5 Data

Go back to fan's Python Code Examples Repository (bookdown site).

3.1.1.1 Basic Conda Setup

1. Download Anaconda for Python 3. For more involved conda instructions see here
2. Open up *anaconda prompt* with admin rights (right click choose as admin).

```
# Inside anaconda prompt
where python
where anaconda
# C:/ProgramData/Anaconda3/Scripts/anaconda.exe
# C:/ProgramData/Anaconda3/python.exe
```

3. Add to Path
4. Install cdsapi and eccodes

```
conda config --add channels conda-forge
conda install -c anaconda pandas
conda install -c conda-forge eccodes -y
conda install -c conda-forge cfgrib -y
conda install -c anaconda xarray
```

3.1.1.2 Account Registration

1. Register for an account
2. Agree to Licence
3. Go to your CDS user page copy the url and key: Get url and key
   - this has UID, 4XXXX, and API KEY, 4XXXfXXX-XXXf-4XXX-9XXX-7XXXebXXfdXX
   - together they should look like: 4XXXX:4XXXfXXX-XXXf-4XXX-9XXX-7XXXebXXfdXX
4. Open up an editor (notepad++ for example), create an empty file, paste the url and your UID:APIKEY into the file as below. Save file as: *C:/Users/fan/.cdsapirc*. Under user root, as *.cdsapirc* file. Note *.cdsapirc* is the file name, you are saving that under the directory *C:/Users/fan/*.

```
url: https://cds.climate.copernicus.eu/api/v2
key: 4XXXX:4XXXfXXX-XXXf-4XXX-9XXX-7XXXebXXfdXX
```

3.1.1.3 Run API Request via Jupyter Notebook

1. open up Jupyter Notebook (this opens up a browser page)
   - cd "C:/Users/fan/Downloads"
   - jupyter notebook
2. create a new *python3* file somewhere you like
3. name the file *cdstest* (saved as ipynb file)
4. paste the code below inside the *ipynb* file you opened (modify *spt_root*):

```
import cdsapi
import urllib.request
# download folder
spt_root = "C:/Users/fan/downloads/_data/"
spn_dl_test_grib = spt_root + "test_china_temp.grib"
# request
c = cdsapi.Client()
res = c.retrieve("reanalysis-era5-pressure-levels",
  {
    'product_type': 'reanalysis',
    'variable': 'temperature',
    'pressure_level': '1000',
    'year': '2008',
    'month': '01',
    'day': '01',
    'time': '12:00',
    'format': 'netcdf',
    'area'           : [53.31, 73, 4.15, 135],
    'grid'           : [1.0, 1.0],
    "format": "grib"
  },
  spn_dl_test_grib
)
# show results
print('print results')
print(res)
print(type(res))
```

5. click run

3.1.1.4 Run API request via Ipython

1. In Anaconda Prompt: *ipython*
2. Open a file in notepad++ or elsewhere, copy the code above over and edit the spt_root to reflect your directories
3. Select the entire code in the notepad++ page, and copy all lines
4. Now inside ipython, type percentage and paste: %paste
5. This should run the file above and save the grib file in the folder you specified with the name you specified.

3.1.1.5 Convert CRIB data to CSV

1. inside conda prompt cd into the folder where you downloaded the grib file
2. *grib_ls* shows what is in the grib file
3. *grib_get_data* translates grib to csv

```
cd "C:/Users/fan/downloads/_data/"
grib_ls test_china_temp.grib
grib_get_data test_china_temp.grib > test_china_temp_raw.csv
```

3.1.1.6 More Advanced Download Setup and Instructions

3.1.1.6.1 Conda Enviornment and Installation In conda, set up a conda environment for downloading ECMWF data using the ECMWF API. (Conda Set-up)

```
# Set up
conda deactivate
conda list env
conda env remove -n wk_ecmwf
conda create -n wk_ecmwf -y
conda activate wk_ecmwf

# Add conda-forge to channel in env
conda config --env --add channels conda-forge
conda config --get channels
conda config --get channels --env

# Install
conda install cdsapi -y
conda install -c anaconda pandas
conda install -c conda-forge eccodes -y
conda install -c conda-forge cfgrib -y
conda install -c anaconda xarray
```

This creates the conda env that we are using here for python.

3.1.1.6.2 Config File .cdsapirc Open up the *cdsapirc*, create new if does note exist. Below, open up the file and save the text. See Python Reading and Writing to File Examples.

First, get the text for the config file:

```
stf_cds_cdsapirc = """\
url: https://cds.climate.copernicus.eu/api/v2
key: 4XXXX:4XXXfXXX-XXXf-4XXX-9XXX-7XXXebXXfdXX\
"""
print(stf_cds_cdsapirc)
```

Second save text to file:

```
# Relative file name
spt_file_cds = "C:/Users/fan/"
snm_file_cds = ".cdsapirc"
spn_file_cds = spt_file_cds + snm_file_cds
# Open new file
fl_cdsapirc_contents = open(spn_file_cds, 'w')
# Write to File
fl_cdsapirc_contents.write(stf_cds_cdsapirc)
# Close
fl_cdsapirc_contents.close()
```

```
# Open the config file to check
code "C:/Users/fan/.cdsapirc"
```

3.1.1.7 Generate API Requests

Go to the sites below, choose download data, pick what is needed, and then select *Show API request* at the bottom of page:

ERA5 pressure levels from 1979 to present

- ERA5 hourly pressure
- ERA5 monthly pressure

ERA5 single levels from 1979 to present

- ERA5 hourly pressure
- ERA5 monthly pressure

3.1.1.7.1 API Request China Temp Test

API function is here.

Select based on China's area, some testing data and download grib file. The data is from 2008, Jan 1st, at 12 noon?

Open up Jupyter notebook: *jupyter notebook*

```
# import module in conda env wk_ecmwf
import cdsapi
import urllib.request

# download folder
spt_root = "C:/Users/fan/pyfan/vig/getdata/envir/"
spn_dl_test_grib = spt_root + "_data/test/test_china_temp.grib"

# request
c = cdsapi.Client()
res = c.retrieve("reanalysis-era5-pressure-levels",
  {
    'product_type': 'reanalysis',
    'variable': 'temperature',
    'pressure_level': '1000',
    'year': '2008',
    'month': '01',
    'day': '01',
    'time': '12:00',
    'format': 'netcdf',
    'area'          : [53.31, 73, 4.15, 135],
    'grid'          : [1.0, 1.0],
    "format": "grib"
  },
  spn_dl_test_grib
)
# show results
print('print results')
print(res)
print(type(res))

# download
# response = urllib.request.urlopen('http://www.example.com/')
# html = response.read()
```

2020-06-17 23:51:35,107 INFO Welcome to the CDS 2020-06-17 23:51:35,107 INFO Sending request to https://cds.climate.copernicus.eu/api/v2/resources/reanalysis-era5-pressure-levels 2020-06-17 23:51:36,441 INFO Request is queued 2020-06-17 23:51:39,183 INFO Request is running 2020-06-17 23:51:45,059 INFO Request is completed 2020-06-17 23:51:45,060 INFO Downloading > http://136.156.133.25/cache-compute-0008/cache/data2/adaptor.mars.internal-1592455900.8655114-11162-11-68e1ea23-8985-4926-95e6-9f181cc7792> 7.grib to C:/Users/fan/pyfan/vig/getdata/envir/_data/test/test_china_temp.grib (6.3K) 2020-06-17 23:51:45,441 INFO Download rate 16.6K/s print results Result(content_length=6480,content_type=application/x-grib,location=http://136.156.133.25/cache-compute-0008/cache/data2/adaptor.mars.inte>

rnal-1592455900.8655114-11162-11-68e1ea23-8985-4926-95e6-9f181cc77927.grib) <class 'cdsapi.api.Result'>

Convert grib to raw csv, open up command line:

```
cd "C:/Users/fan/pyfan/vig/getdata/envir/_data/test/"
grib_ls test_china_temp.grib
grib_get_data test_china_temp.grib > test_china_temp_raw.csv
```

Format the CSV file (is not comma separated)

```
spt_root = "C:/Users/fan/pyfan/vig/getdata/envir/_data/test/"
spn_csv_raw = spt_root + "test_china_temp_raw.csv"
spn_csv_edi = spt_root + "test_china_temp.csv"

with open(spn_csv_raw, 'r') as f_in, open(spn_csv_edi, 'w') as f_out:
    f_out.write(next(f_in))
    [f_out.write(','.join(line.split()) + '\n') for line in f_in]
```

Show CSV results:

```
# Path and Read
spt_root = "C:/Users/fan/pyfan/vig/getdata/envir/"
spn_dl_test_csv = paste0(spt_root, "_data/test/test_china_temp.csv")
china_weather_data <- read.csv(spn_dl_test_csv)

# Top 50 rows
kable(head(china_weather_data, 50),
      caption="Chinese Long and Lat, Temperature Pressure, 2008 Jan 1st at 12 noon?") %>%
  kable_styling_fc()
```

Chinese Long and Lat, Temperature Pressure, 2008 Jan 1st at 12 noon?

time	latitude	longitude	u10	v10	d2m	t2m	msl	sp
2008-01-01 02:00:00	23.25	113	-2.6031342	-4.829605	265.4314	284.8363	102918.8	102616.0
2008-01-01 02:00:00	23.25	114	-2.7173920	-3.808121	262.7693	284.2719	102862.1	101628.0
2008-01-01 02:00:00	22.25	113	-2.6246185	-6.311050	266.9294	284.2602	102796.6	102059.0
2008-01-01 02:00:00	22.25	114	-2.6285248	-6.152847	267.4978	285.3168	102710.1	102201.0
2008-01-01 12:00:00	23.25	113	-1.1495056	-2.728592	265.8091	286.1729	102588.9	102290.7
2008-01-01 12:00:00	23.25	114	-1.4454040	-2.477615	265.3033	285.0987	102591.6	101374.7
2008-01-01 12:00:00	22.25	113	-0.6924744	-4.270584	268.0396	286.5753	102482.9	101757.7
2008-01-01 12:00:00	22.25	114	-1.9668884	-4.906326	266.7486	288.0030	102440.1	101940.7

"ERA5 is a comprehensive reanalysis, from 1979 (soon to be backdated to 1950) to near real time, which assimilates as many observations as possible in the upper air and near surface. The ERA5 atmospheric model is coupled with a land surface model and a wave model."

1. Register for an account
2. Agree to Licence

3.1.1.8 Learning

3.1.1.8.1 Terminologies Links:

- status of the CDS queue.

Terminologies:

- single level parameters

3.1.1.8.2 Single Level Parameters ERA5 Variables?

1. Table 1: surface and single level parameters: invariants

2. Table 9: pressure level parameters: instantaneous

- Temperature

ER5 Data Download Instructions.

3.1.1.9 UTCI, NC Format Data, Download, Unzip, Convert to combined CSV

The data downloaded from CDS climate could become very large in size. We want to process parts of the data one part at a time, summarize and aggregate over each part, and generate a file output file with aggregate statistics over the entire time period of interest.

This code below accompalishes the following tasks:

1. download data from derived-utci-historical as ZIP: API request by itself
2. unzip
3. convert *nc* files to *csv* files
4. individual csv files are half year groups

Parameter Control for the code below:

1. *spt_root*: root folder where everything will be at
2. *spth_conda_env*: the conda virtual environment python path, eccodes and cdsapi packages are installed in the conda virtual environment. In the example below, the first env is: wk_ecmwf
3. *st_nc_prefix*: the downloaded individual nc files have dates and prefix before and after the date string in the nc file names. This is the string before that.
4. *st_nc_suffix*: see (3), this is the suffix
5. *ar_years*: array of years to download and aggregate over
6. *ar_months_g1*: months to download in first half year
7. *ar_months_g2*: months to download in second half year

```
################################################
# ------------ Parameters
################################################

# Where to store everything
spt_root <- "C:/Users/fan/Downloads/_data/"
spth_conda_env <- "C:/ProgramData/Anaconda3/envs/wk_ecmwf/python.exe"
# nc name prefix
st_nc_prefix <- "ECMWF_utci_"
st_nc_suffix <- "_v1.0_con.nc"
# Years list
# ar_years <- 2001:2019
ar_years <- c(2005, 2015)
# ar_months_g1 <- c('01','02','03','04','05','06')
ar_months_g1 <- c('01', '03')
# ar_months_g2 <- c('07','08','09','10','11','12')
ar_months_g2 <- c('07', '09')


# folder to download any nc zips to
nczippath <- spt_root
# we are changing the python api file with different requests stirngs and storing it here
pyapipath <- spt_root
# output directory for AGGREGATE CSV with all DATES from this search
csvpath <- spt_root

################################################
# ------------ Packages
################################################

library("ncdf4")
```

```
library("chron")
library("lattice")
library("RColorBrewer")
library("stringr")
library("tibble")
library("dplyr")
Sys.setenv(RETICULATE_PYTHON = spth_conda_env)
library("reticulate")

################################################
# ------------ Define Loops
################################################
for (it_yr in ar_years) {
  for (it_mth_group in c(1,2)) {
    if(it_mth_group == 1) {
      ar_months = ar_months_g1
    }
    if(it_mth_group == 2) {
      ar_months = ar_months_g2
    }

    ################################################
    # ------------ Define Python API Call
    ################################################

    # name of zip file
    nczipname <- "derived_utci_2010_2.zip"
    unzipfolder <- "derived_utci_2010_2"

    st_file <- paste0("import cdsapi
import urllib.request
# download folder
spt_root = '", nczippath, "'
spn_dl_test_grib = spt_root + '", nczipname, "'
# request
c = cdsapi.Client()
res = c.retrieve(
    'derived-utci-historical',
    {
        'format': 'zip',
        'variable': 'Universal thermal climate index',
        'product_type': 'Consolidated dataset',
        'year': '",it_yr, "',
        'month': [
            ", paste("'", ar_months, "'", sep = "", collapse = ", "), "
        ],
        'day': [
            '01','03'
        ],
        'area'  : [53.31, 73, 4.15, 135],
        'grid'  : [0.25, 0.25],
    },
    spn_dl_test_grib)
# show results
print('print results')
print(res)
print(type(res))")
```

```
# st_file = "print(1+1)"

# Store Python Api File
fl_test_tex <- paste0(pyapipath, "api.py")
fileConn <- file(fl_test_tex)
writeLines(st_file, fileConn)
close(fileConn)

################################################
# ------------ Run Python File
################################################
# Set Path
setwd(pyapipath)
# Run py file, api.py name just defined
use_python(spth_conda_env)
source_python('api.py')

################################################
# ------------ uNZIP
################################################
spn_zip <- paste0(nczippath, nczipname)
spn_unzip_folder <- paste0(nczippath, unzipfolder)
unzip(spn_zip, exdir=spn_unzip_folder)

################################################
# ------------ Find All files
################################################
# Get all files with nc suffix in folder
ncpath <- paste0(nczippath, unzipfolder)
ls_sfls <- list.files(path=ncpath, recursive=TRUE, pattern=".nc", full.names=T)

################################################
# ------------ Combine individual NC files to JOINT Dataframe
################################################
# List to gather dataframes
ls_df <- vector(mode = "list", length = length(ls_sfls))
# Loop over files and convert nc to csv
it_df_ctr <- 0
for (spt_file in ls_sfls) {
  it_df_ctr <- it_df_ctr + 1

  # Get file name without Path
  snm_file_date <- sub(paste0('\\',st_nc_suffix,'$'), '', basename(spt_file))
  snm_file_date <- sub(st_nc_prefix, '', basename(snm_file_date))

  # Dates Start and End: list.files is auto sorted in ascending order
  if (it_df_ctr == 1) {
    snm_start_date <- snm_file_date
  }
  else {
    # this will give the final date
    snm_end_date <- snm_file_date
  }

  # Given this structure: ECMWF_utci_20100702_v1.0_con, sub out prefix and suffix
  print(spt_file)
  ncin <- nc_open(spt_file)
```

```
  nchist <- ncatt_get(ncin, 0, "history")

  # not using this missing value flag at the moment
  missingval <- str_match(nchist$value, "setmisstoc,\\s*(.*?)\\s* ")[,2]
  missingval <- as.numeric(missingval)

  lon <- ncvar_get(ncin, "lon")
  lat <- ncvar_get(ncin, "lat")
  tim <- ncvar_get(ncin, "time")
  tunits <- ncatt_get(ncin, "time", "units")

  nlon <- dim(lon)
  nlat <- dim(lat)
  ntim <- dim(tim)

  # convert time -- split the time units string into fields
  # tustr <- strsplit(tunits$value, " ")
  # tdstr <- strsplit(unlist(tustr)[3], "-")
  # tmonth <- as.integer(unlist(tdstr)[2])
  # tday <- as.integer(unlist(tdstr)[3])
  # tyear <- as.integer(unlist(tdstr)[1])
  # mytim <- chron(tim, origin = c(tmonth, tday, tyear))

  tmp_array <- ncvar_get(ncin, "utci")
  tmp_array <- tmp_array - 273.15

  lonlat <- as.matrix(expand.grid(lon = lon, lat = lat, hours = tim))
  temperature <- as.vector(tmp_array)
  tmp_df <- data.frame(cbind(lonlat, temperature))

  # extract a rectangle
  eps <- 1e-8
  minlat <- 22.25 - eps
  maxlat <- 23.50 + eps
  minlon <- 113.00 - eps
  maxlon <- 114.50 + eps
  # subset data
  subset_df <- tmp_df[tmp_df$lat >= minlat & tmp_df$lat <= maxlat &
                        tmp_df$lon >= minlon & tmp_df$lon <= maxlon, ]

  # add Date
  subset_df_date <- as_tibble(subset_df) %>% mutate(date = snm_file_date)

  # Add to list
  ls_df[[it_df_ctr]] <- subset_df_date

  # Close NC
  nc_close(ncin)
}

# List of DF to one DF
df_all_nc <- do.call(rbind, ls_df)

# Save File
fname <- paste0(paste0(st_nc_prefix,
                       snm_start_date, "_to_", snm_end_date,
                       ".csv"))
```

```
    csvfile <- paste0(csvpath, fname)
    write.table(na.omit(df_all_nc), csvfile, row.names = FALSE, sep = ",")

    # Delete folders
    unlink(spn_zip, recursive=TRUE, force=TRUE)
    unlink(spn_unzip_folder, recursive=TRUE, force=TRUE)

  # end loop months groups
  }
# end loop year
}
```

Chapter 4

System and Support

4.1 Command Line

4.1.1 Python Command Line

Go back to fan's Python Code Examples Repository (bookdown site).

4.1.1.1 Run Command Line from Inside Python

Use subprocess, where is python:

```
import subprocess
cmd_popen = subprocess.Popen(["where", "python"],
                             stdin=subprocess.PIPE,
                             stdout=subprocess.PIPE,
                             stderr=subprocess.PIPE)
output, err = cmd_popen.communicate()
print(output.decode('utf-8'))
```

```
## C:\ProgramData\Anaconda3\envs\wk_pyfan\python.exe
## C:\ProgramData\Anaconda3\python.exe
## C:\Users\fan\.windows-build-tools\python27\python.exe
## C:\Users\fan\AppData\Local\Microsoft\WindowsApps\python.exe
## C:\Program Files\Inkscape\python.exe
```

Command line file redirection symbol,

```
# The > command line sends current console output to file.txt
# cd "C:\users\fan"
# ls > ls_files.txt
# rm ls_files.txt
import os
import subprocess

# ls in current location
cmd_ls = subprocess.Popen(["ls"], stdin=subprocess.PIPE, stdout=subprocess.PIPE, stderr=subprocess.P
stf_out_cmd_ls, err = cmd_ls.communicate()
stf_out_cmd_ls = stf_out_cmd_ls.decode('utf-8')
print(stf_out_cmd_ls)
```

```
## A Collection of Python Examples.Rmd
## A-Collection-of-Python-Examples.Rmd
## A-Collection-of-Python-Examples_files
## LICENSE
## README.md
```

```
## README_appendix.md
## README_end.md
## README_pre.md
## README_toc.md
## _bookdown.yml
## _bookdown_files
## _config.yml
## _file
## _folder
## _log
## _m
## _output.yml
## _output_kniti_html.yaml
## _output_kniti_pdf.yaml
## _templates
## book.bib
## bookdown
## build
## dist
## doc
## hdga.html
## htmlpdfr
## img
## index.Rmd
## packages.bib
## poetry.lock
## preamble.tex
## preamble_book.tex
## pyfan
## pyfan.egg-info
## pyproject.toml
## setup.py
## style.css
## tests
## title.Rmd
## vig
```

```
srt_file_tex = "_file/"
sna_file_tex = "test_ls_pyfanvig_stdout"
srn_file_tex = srt_file_tex + sna_file_tex + ".txt"
fl_tex_contents = open(srn_file_tex, 'w')
fl_tex_contents.write(stf_out_cmd_ls)
```

```
## 520
```

```
fl_tex_contents.close()
```

4.1.1.2 Execute Command Line Python Functions

- run python from command line
- run python function with parameters from command line

Here run python from command line inside python itself.

```
# Run:
from py.fan.util.rmd.mattexmd import fp_mlxtex2md
fp_mlxtex2md(spt_root='C:/Users/fan/Math4Econ/matrix_application/', ls_srt_subfolders=None, st_rglob
```

```
# Run:
python -c "from pyfan.util.rmd.mattexmd import fp_mlxtex2md; fp_mlxtex2md(spt_root='C:/Users/fan/Mat
```

4.1.2 Run Matlab Functions

Go back to fan's Python Code Examples Repository (bookdown site).

4.1.2.1 Generate A template Matlab Script

Generate an example matlab script file with parameter x.

```
# Example Matlab Function
stf_m_contents = """\
a = x + 1
b = 10*x\
"""
# Print
print(stf_m_contents)
# Open new file
```

```
## a = x + 1
## b = 10*x
```

```
fl_m_contents = open("_m/fs_test.m", 'w')
# Write to File
fl_m_contents.write(stf_m_contents)
# print
```

```
## 18
```

```
fl_m_contents.close()
```

4.1.2.2 Run the Matlab Function from Commandline

- run matlab function from command line
- Retrieving the output of subprocess.call
- https://www.mathworks.com/help/matlab/ref/matlabwindows.html

First, check where matlab is installed:

```
import subprocess
cmd_popen = subprocess.Popen(["where", "matlab"],
                             stdin=subprocess.PIPE,
                             stdout=subprocess.PIPE,
                             stderr=subprocess.PIPE)
output, err = cmd_popen.communicate()
print(output.decode('utf-8'))
```

```
## C:\Program Files\MATLAB\R2019b\bin\matlab.exe
```

Second, run the matlab file, first definet he parameter x:

```
import os
# print and set directory
print(os.getcwd())
```

```
## C:\Users\fan\pyfan
```

```
os.chdir('_m')
print(os.getcwd())
# run matlab script saved prior
# running command line: matlab -batch "fs_test; exit"
```

```
## C:\Users\fan\pyfan\_m
```

```
cmd_popen = subprocess.Popen(["matlab", "-batch", "\"x=1; fs_test; exit\""],
                             stdin=subprocess.PIPE,
                             stdout=subprocess.PIPE,
```

```
                              stderr=subprocess.PIPE)
output, err = cmd_popen.communicate()
print(output.decode('utf-8'))
```

```
##
## a =
##
##      2
##
##
## b =
##
##     10
##
```

Third, run the function again, but with *x=3*:

```
os.chdir('_m')
print(os.getcwd())
```

```
## C:\Users\fan\pyfan\_m
```

```
print(subprocess.Popen(["matlab", "-batch", "\"x=5; fs_test; exit\""],
                       stdin=subprocess.PIPE,
                       stdout=subprocess.PIPE,
                       stderr=subprocess.PIPE).communicate()[0].decode('utf-8'))
```

```
##
## a =
##
##      6
##
##
## b =
##
##     50
##
```

4.2 File In and Out

4.2.1 Read and Write and Convert

Go back to fan's Python Code Examples Repository (bookdown site).

- python create a text file
- python write file from paragraphs

4.2.1.1 Generate a tex file

Will a bare-bone tex file with some texts inside, save inside the *_file* subfolder.

First, create the text text string, note the the linebreaks utomatically generate linebreaks, note that slash need double slash:

```
# Create the Tex Text
# Note that trible quotes begin first and end last lines
stf_tex_contents = """\\documentclass[12pt,english]{article}
\\usepackage[bottom]{footmisc}
\\usepackage[urlcolor=blue]{hyperref}
\\begin{document}
\\title{A Latex Testing File}
```

```
\\author{\\href{http://fanwangecon.github.io/}{Fan Wang} \\thanks{See information \\href{https://fan
\\maketitle
Ipsum information dolor sit amet, consectetur adipiscing elit. Integer Latex placerat nunc orci.
\\paragraph{\\href{https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3140132}{Data}}
Village closure information is taken from a village head survey.\\footnote{Generally students went t
\\end{document}"""
# Print
print(stf_tex_contents)
```

```
## \documentclass[12pt,english]{article}
## \usepackage[bottom]{footmisc}
## \usepackage[urlcolor=blue]{hyperref}
## \begin{document}
## \title{A Latex Testing File}
## \author{\href{http://fanwangecon.github.io/}{Fan Wang} \thanks{See information \href{https://fanw
## \maketitle
## Ipsum information dolor sit amet, consectetur adipiscing elit. Integer Latex placerat nunc orci.
## \paragraph{\href{https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3140132}{Data}}
## Village closure information is taken from a village head survey.\footnote{Generally students went
## \end{document}
```

Second, write the contents of the file to a new tex file stored inside the *_file* subfolder of the directory:

```
# Relative file name
srt_file_tex = "_file/"
sna_file_tex = "test_fan"
srn_file_tex = srt_file_tex + sna_file_tex + ".tex"
# Open new file
fl_tex_contents = open(srn_file_tex, 'w')
# Write to File
fl_tex_contents.write(stf_tex_contents)
# print
```

```
## 617
```

```
fl_tex_contents.close()
```

4.2.1.2 Replace Strings in a tex file

Replace a set of strings in the file just generated by a set of alternative strings.

```
# Open file Get text
fl_tex_contents = open(srn_file_tex)
stf_tex_contents = fl_tex_contents.read()
print(srn_file_tex)

# define new and old
```

```
## _file/test_fan.tex
```

```
ls_st_old = ['information', 'Latex']
ls_st_new = ['INFOREPLACE', 'LATEX']

# zip and loop and replace
for old, new in zip(ls_st_old, ls_st_new):
  stf_tex_contents = stf_tex_contents.replace(old, new)
print(stf_tex_contents)

# write to file with replacements
```

```
## \documentclass[12pt,english]{article}
```

```
## \usepackage[bottom]{footmisc}
## \usepackage[urlcolor=blue]{hyperref}
## \begin{document}
## \title{A LATEX Testing File}
## \author{\href{http://fanwangecon.github.io/}{Fan Wang} \thanks{See INFOREPLACE \href{https://fanw
## \maketitle
## Ipsum INFOREPLACE dolor sit amet, consectetur adipiscing elit. Integer LATEX placerat nunc orci.
## \paragraph{\href{https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3140132}{Data}}
## Village closure INFOREPLACE is taken from a village head survey.\footnote{Generally students went
## \end{document}
```

```
sna_file_edited_tex = "test_fan_edited"
srn_file_edited_tex = srt_file_tex + sna_file_edited_tex + ".tex"
fl_tex_ed_contents = open(srn_file_edited_tex, 'w')
fl_tex_ed_contents.write(stf_tex_contents)
```

```
## 617
```

```
fl_tex_ed_contents.close()
```

4.2.1.3 Convert Tex File to Pandoc and Compile Latex

Compile tex file to pdf and clean up the extraneous pdf outputs. See ff_pdf_gen_clean.

```
import subprocess
import os

# Change to local directory so path in tex respected.
os.chdir("C:/Users/fan/pyfan/vig/support/inout")

# Convert tex to pdf
subprocess.call(['C:/texlive/2019/bin/win32/xelatex.exe', '-output-directory',
                 srt_file_tex, srn_file_edited_tex], shell=False)
# Clean pdf extraneous output
```

```
## 0
```

```
ls_st_remove_suffix = ['aux','log','out','bbl','blg']
for st_suffix in ls_st_remove_suffix:
    srn_cur_file = srt_file_tex + sna_file_edited_tex + "." + st_suffix
    if (os.path.isfile(srn_cur_file)):
        os.remove(srt_file_tex + sna_file_edited_tex + "." + st_suffix)
```

Use pandoc to convert tex file

```
import subprocess

# md file name
srn_file_md = srt_file_tex + "test_fan_edited.md"
# Convert tex to md
subprocess.call(['pandoc', '-s', srn_file_tex, '-o', srn_file_md])
# Open md file
```

```
## 0
```

```
fl_md_contents = open(srn_file_md)
print(fl_md_contents.read())
```

```
## ---
## author:
## - '[Fan Wang](http://fanwangecon.github.io/) [^1]'
## title: A Latex Testing File
## ---
```

```
##
## \maketitle
## Ipsum information dolor sit amet, consectetur adipiscing elit. Integer
## Latex placerat nunc orci.
##
## #### [Data](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3140132)
##
## Village closure information is taken from a village head survey.[^2]
##
## [^1]: See information
##     [Tex4Econ](https://fanwangecon.github.io/Tex4Econ/) for more.
##
## [^2]: Generally students went to schools.
```

4.2.1.4 Search for Files with Suffix in Several Folders

- python search all files in folders with suffix

Search for files in several directories that have a particular suffix. Then decompose directory into sub-components.

Search file inside several folders (not recursively in subfolders):

```
from pathlib import Path

# directories to search in
ls_spt_srh = ["C:/Users/fan/R4Econ/amto/",
              "C:/Users/fan/R4Econ/function/"]

# get file names in folders (not recursively)
ls_spn_found = [spn_file for spt_srh in ls_spt_srh
                         for spn_file in Path(spt_srh).glob('*.Rmd')]
for spn_found in ls_spn_found:
  print(spn_found)
```

```
## C:\Users\fan\R4Econ\amto\main.Rmd
## C:\Users\fan\R4Econ\function\main.Rmd
```

Search file recursivesly in all subfolders of folders:

```
from pathlib import Path

# directories to search in
ls_spt_srh = ["C:/Users/fan/R4Econ/amto/array/",
              "C:/Users/fan/R4Econ/amto/list"]

# get file names recursively in all subfolders
ls_spn_found = [spn_file for spt_srh in ls_spt_srh
                         for spn_file in Path(spt_srh).rglob('*.R')]
for spn_found in ls_spn_found:
  drive, path_and_file = os.path.splitdrive(spn_found)
  path_no_suffix = os.path.splitext(spn_found)[0]
  path_no_file, file = os.path.split(spn_found)
  file_no_suffix = Path(spn_found).stem
  print('file:', file, '\ndrive:', drive,
        '\nfile no suffix:', file_no_suffix,
        '\nfull path:', spn_found,
        '\npt no fle:', path_no_file,
        '\npt no suf:', path_no_suffix, '\n')
```

```
## file: fs_ary_basics.R
```

```
## drive: C:
## file no suffix: fs_ary_basics
## full path: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_basics.R
## pt no fle: C:\Users\fan\R4Econ\amto\array\htmlpdfr
## pt no suf: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_basics
##
## file: fs_ary_generate.R
## drive: C:
## file no suffix: fs_ary_generate
## full path: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_generate.R
## pt no fle: C:\Users\fan\R4Econ\amto\array\htmlpdfr
## pt no suf: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_generate
##
## file: fs_ary_mesh.R
## drive: C:
## file no suffix: fs_ary_mesh
## full path: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_mesh.R
## pt no fle: C:\Users\fan\R4Econ\amto\array\htmlpdfr
## pt no suf: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_mesh
##
## file: fs_ary_string.R
## drive: C:
## file no suffix: fs_ary_string
## full path: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_string.R
## pt no fle: C:\Users\fan\R4Econ\amto\array\htmlpdfr
## pt no suf: C:\Users\fan\R4Econ\amto\array\htmlpdfr\fs_ary_string
##
## file: fs_listr.R
## drive: C:
## file no suffix: fs_listr
## full path: C:\Users\fan\R4Econ\amto\list\htmlpdfr\fs_listr.R
## pt no fle: C:\Users\fan\R4Econ\amto\list\htmlpdfr
## pt no suf: C:\Users\fan\R4Econ\amto\list\htmlpdfr\fs_listr
##
## file: fs_lst_basics.R
## drive: C:
## file no suffix: fs_lst_basics
## full path: C:\Users\fan\R4Econ\amto\list\htmlpdfr\fs_lst_basics.R
## pt no fle: C:\Users\fan\R4Econ\amto\list\htmlpdfr
## pt no suf: C:\Users\fan\R4Econ\amto\list\htmlpdfr\fs_lst_basics
```

4.2.2 Folder Operations

Go back to fan's Python Code Examples Repository (bookdown site).

4.2.2.1 New Folder and Files

1. create a folder and subfolder
2. create two files in the new folder

```
import pathlib

# folder root
srt_folder = "_folder/"

# new folder
srt_subfolder = srt_folder + "fa/"
# new subfolder
srt_subfolder = srt_subfolder + "faa/"
```

```
# generate folders recursively
pathlib.Path(srt_subfolder).mkdir(parents=True, exist_ok=True)

# Open new file
fl_tex_contents_aa = open(srt_subfolder + "file_a.txt", 'w')
# Write to File
fl_tex_contents_aa.write('contents of file a')
```

```
## 18
```

```
fl_tex_contents_aa.close()

# Open another new file and save
fl_tex_contents_ab = open(srt_subfolder + "file_b.txt", 'w')
# Write to File
fl_tex_contents_ab.write('contents of file b')
```

```
## 18
```

```
fl_tex_contents_ab.close()
```

Generate more folders without files:

```
# generate folders recursively
pathlib.Path("_folder/fb/fba/").mkdir(parents=True, exist_ok=True)
# generate folders recursively
pathlib.Path("_folder/fc/").mkdir(parents=True, exist_ok=True)
# generate folders recursively
pathlib.Path("_folder/fd/").mkdir(parents=True, exist_ok=True)
```

4.2.2.2 Copy a File from One Folder to Another

Move the two files from *_folder/fa/faa* to *_folder/faa* as well as to *_folder/fb/faa. *Use* shutil.copy2* so that more metadata is copied over. But *copyfile* is faster.

- How do I copy a file in Python?

Moving one file:

```
import shutil
# Faster method
shutil.copyfile('_folder/fa/faa/file_a.txt', '_folder/fb/file_a.txt')
# More metadat copied, and don't need to specify name
```

```
## '_folder/fb/file_a.txt'
```

```
shutil.copy2('_folder/fa/faa/file_a.txt', '_folder/fb/fba')
```

```
## '_folder/fb/fba\\file_a.txt'
```

4.2.2.3 Copy Folder to Multiple Destimations

Move Entire Folder, How do I copy an entire directory of files into an existing directory using Python?:

```
from distutils.dir_util import copy_tree

# Move contents from fa/faa/ to fc/faa
srt_currroot = '_folder/fa/'
srt_folder = 'faa/'
srt_newroot = '_folder/fc/'

# Full source and destination
srt_sourc = srt_currroot + srt_folder
```

```
srt_desct = srt_newroot + srt_folder

# Check/Create new Directory
pathlib.Path(srt_desct).mkdir(parents=True, exist_ok=True)

# Move
copy_tree(srt_sourc, srt_desct)
```

```
## ['_folder/fc/faa/file_a.txt', '_folder/fc/faa/file_b.txt']
```

Move contents to multiple destinations:

```
from distutils.dir_util import copy_tree
# Check/Create new Directory
pathlib.Path('_folder/fd/faa/fa_images').mkdir(parents=True, exist_ok=True)
pathlib.Path('_folder/fd/faa/fb_images').mkdir(parents=True, exist_ok=True)
pathlib.Path('_folder/fd/faa/fc_images').mkdir(parents=True, exist_ok=True)
pathlib.Path('_folder/fd/faa/fz_img').mkdir(parents=True, exist_ok=True)
pathlib.Path('_folder/fd/faa/fz_other').mkdir(parents=True, exist_ok=True)

# Move
copy_tree('_folder/fa/faa/', '_folder/fd/faa/fa_images')
```

```
## ['_folder/fd/faa/fa_images\\file_a.txt', '_folder/fd/faa/fa_images\\file_b.txt']
```

```
copy_tree('_folder/fa/faa/', '_folder/fd/faa/fb_images')
```

```
## ['_folder/fd/faa/fb_images\\file_a.txt', '_folder/fd/faa/fb_images\\file_b.txt']
```

```
copy_tree('_folder/fa/faa/', '_folder/fd/faa/fc_images')
```

```
## ['_folder/fd/faa/fc_images\\file_a.txt', '_folder/fd/faa/fc_images\\file_b.txt']
```

```
copy_tree('_folder/fa/faa/', '_folder/fd/faa/fz_img')
```

```
## ['_folder/fd/faa/fz_img\\file_a.txt', '_folder/fd/faa/fz_img\\file_b.txt']
```

```
copy_tree('_folder/fa/faa/', '_folder/fd/faa/fz_other')
# Empty Folder
```

```
## ['_folder/fd/faa/fz_other\\file_a.txt', '_folder/fd/faa/fz_other\\file_b.txt']
```

```
pathlib.Path('_folder/fd/faa/fd_images').mkdir(parents=True, exist_ok=True)
pathlib.Path('_folder/fd/faa/fe_images').mkdir(parents=True, exist_ok=True)
```

4.2.2.4 Search for Files in Folder

Find the total number of files in a folder.

```
import pathlib
# the number of files in folder found with search critiera
st_fle_search = '*.txt'
ls_spn = [Path(spn).stem for spn in Path('_folder/fd/faa/fa_images').rglob(st_fle_search)]
print(ls_spn)

# count files in a non-empty folder
```

```
## ['file_a', 'file_b']
```

```
srn = '_folder/fd/faa/fa_images'
[spn for spn in Path(srn).rglob(st_fle_search)]
```

```
## [WindowsPath('_folder/fd/faa/fa_images/file_a.txt'), WindowsPath('_folder/fd/faa/fa_images/file_b
```

```
bl_folder_is_empty = len([spn for spn in Path(srn).rglob(st_fle_search)])>0
print(bl_folder_is_empty)

# count files in an empty folder
```

```
## True
```

```
srn = '_folder/fd/faa/fd_images'
[spn for spn in Path(srn).rglob(st_fle_search)]
```

```
## []
```

```
bl_folder_is_empty = len([spn for spn in Path(srn).rglob(st_fle_search)])>0
print(bl_folder_is_empty)
```

```
## False
```

4.2.2.5 Search for Folder Names

- python search for folders containing strings

Search for folders with certain search word in folder name, and only keep if folder actually has files.

```
import os

# get all folder names in folder
ls_spt = os.listdir('_folder/fd/faa/')
print(ls_spt)

# Select only subfolder names containing _images
```

```
## ['fa_images', 'fb_images', 'fc_images', 'fd_images', 'fe_images', 'fz_img', 'fz_other', '_img']
```

```
srt = '_folder/fd/faa/'
st_search = '_images'
ls_srt_found = [srt + spt
                for spt in os.listdir(srt)
                if st_search in spt]
print(ls_srt_found)
```

```
## ['_folder/fd/faa/fa_images', '_folder/fd/faa/fb_images', '_folder/fd/faa/fc_images', '_folder/fd/
```

4.2.2.6 Find Non-empty Folders by Name

Search:

1. Get subfolders in folder with string in name
2. Only collect if there are files in the subfolder

```
import pathlib

# Select only subfolder names containing _images
srt = '_folder/fd/faa/'
# the folder names must contain _images
st_srt_srh = '_images'
# there must be files in the folder with this string
st_fle_srh = '*.txt'

# All folders that have String
ls_srt_found = [srt + spt
                for spt in os.listdir(srt)
                if st_srt_srh in spt]
print(ls_srt_found)
```

```
# All folders that have String and that are nonempty
```

```
## ['_folder/fd/faa/fa_images', '_folder/fd/faa/fb_images', '_folder/fd/faa/fc_images', '_folder/fd/
```

```
ls_srt_found = [srt + spt
                for spt in os.listdir(srt)
                if ((st_srt_srh in spt)
                    and
                    (len([spn for spn
                          in Path(srt + spt).rglob(st_fle_srh)])>0)) ]
print(ls_srt_found)
```

```
## ['_folder/fd/faa/fa_images', '_folder/fd/faa/fb_images', '_folder/fd/faa/fc_images']
```

4.2.2.7 Found Folders to new Folder

1. Search for subfolders by folder name string in a folder
2. Select nonempty subfolders
3. Move nonsempty subfolders to one new folder
4. Move this single combination folder

The results here are implemented as function in the pyfan package: fp_agg_move_subfiles.

```
import pathlib
import os
import shutil
from distutils.dir_util import copy_tree

# 1 Define Parameters

# Select only subfolder names containing _images
srt = '_folder/fd/faa/'
# the folder names must contain _images
st_srt_srh = '_images'
# there must be files in the folder with this string
st_fle_srh = '*.txt'

# new aggregating folder name
srt_agg = '_img'

# folders to move aggregation files towards
ls_srt_dest = ['_folder/fd/faa/', '_folder/']

# delete source
bl_delete_source = False

# 2 Gather Folders
ls_ls_srt_found = [[srt + spt, spt]
                   for spt in os.listdir(srt)
                   if ((st_srt_srh in spt)
                       and
                       (len([spn for spn
                             in Path(srt + spt).rglob(st_fle_srh)])>0)) ]
print(ls_ls_srt_found)

# 3 Loop over destination folders, loop over source folders
```

```
## [['_folder/fd/faa/fa_images', 'fa_images'], ['_folder/fd/faa/fb_images', 'fb_images'], ['_folder/
```

```
for srt in ls_srt_dest:

  # Move each folder over
  for ls_srt_found in ls_ls_srt_found:

    # Paths
    srt_source = ls_srt_found[0]
    srt_dest = os.path.join(srt, srt_agg, ls_srt_found[1])

    # dest folders
    pathlib.Path(srt_dest).mkdir(parents=True, exist_ok=True)

    # move
    copy_tree(ls_srt_found[0], srt_dest)

# 4. Delete Sources
```

```
## ['_folder/fd/faa/_img\\fa_images\\file_a.txt', '_folder/fd/faa/_img\\fa_images\\file_b.txt']
## ['_folder/fd/faa/_img\\fb_images\\file_a.txt', '_folder/fd/faa/_img\\fb_images\\file_b.txt']
## ['_folder/fd/faa/_img\\fc_images\\file_a.txt', '_folder/fd/faa/_img\\fc_images\\file_b.txt']
## ['_folder/_img\\fa_images\\file_a.txt', '_folder/_img\\fa_images\\file_b.txt']
## ['_folder/_img\\fb_images\\file_a.txt', '_folder/_img\\fb_images\\file_b.txt']
## ['_folder/_img\\fc_images\\file_a.txt', '_folder/_img\\fc_images\\file_b.txt']
```

```
if bl_delete_source:
  for ls_srt_found in ls_ls_srt_found:
    shutil.rmtree(ls_srt_found[0])
```

4.2.3 Parse Yaml

Go back to fan's Python Code Examples Repository (bookdown site).

Use the PyYAML to parse yaml.

4.2.3.1 Write and Create a Simple YAML file

First, Yaml as a string variable:

```
# Create the Tex Text
# Note that trible quotes begin first and end last lines
stf_tex_contents = """\
- file: matrix_matlab
  title: "One Variable Graphs and Tables"
  description: |
    Frequency table, bar chart and histogram.
    R function and lapply to generate graphs/tables for different variables.
  core:
  - package: r
    code: |
      c('word1','word2')
      function()
      for (ctr in c(1,2)) {}
  - package: dplyr
    code: |
      group_by()
  date: 2020-05-02
  output:
    pdf_document:
      pandoc_args: '../_output_kniti_pdf.yaml'
      includes:
```

```
        in_header: '../preamble.tex'
  urlcolor: blue
- file: matrix_algebra_rules
  title: "Opening a Dataset"
  titleshort: "Opening a Dataset"
  description: |
    Opening a Dataset.
  core:
  - package: r
    code: |
      setwd()
  - package: readr
    code: |
      write_csv()
  date: 2020-05-02
  date_start: 2018-12-01
- file: matrix_two
  title: "Third file"
  titleshort: "Third file"
  description: |
    Third file description."""
# Print
print(stf_tex_contents)
```

```
## - file: matrix_matlab
##   title: "One Variable Graphs and Tables"
##   description: |
##     Frequency table, bar chart and histogram.
##     R function and lapply to generate graphs/tables for different variables.
##   core:
##   - package: r
##     code: |
##       c('word1','word2')
##       function()
##       for (ctr in c(1,2)) {}
##   - package: dplyr
##     code: |
##       group_by()
##   date: 2020-05-02
##   output:
##     pdf_document:
##       pandoc_args: '../_output_kniti_pdf.yaml'
##       includes:
##         in_header: '../preamble.tex'
##   urlcolor: blue
## - file: matrix_algebra_rules
##   title: "Opening a Dataset"
##   titleshort: "Opening a Dataset"
##   description: |
##     Opening a Dataset.
##   core:
##   - package: r
##     code: |
##       setwd()
##   - package: readr
##     code: |
##       write_csv()
##   date: 2020-05-02
```

```
##   date_start: 2018-12-01
## - file: matrix_two
##   title: "Third file"
##   titleshort: "Third file"
##   description: |
##     Third file description.
```

Second, write the contents of the file to a new tex file stored inside the *_file* subfolder of the directory:

```
# Relative file name
srt_file_tex = "_file/"
sna_file_tex = "test_yml_fan"
srn_file_tex = srt_file_tex + sna_file_tex + ".yml"
# Open new file
fl_tex_contents = open(srn_file_tex, 'w')
# Write to File
fl_tex_contents.write(stf_tex_contents)
# print
```

```
## 908
```

```
fl_tex_contents.close()
```

4.2.3.2 Select Subset of Values by Key

Load Yaml file created prior, the output is a list of dictionaries:

```
import yaml
import pprint
# Open yaml file
fl_yaml = open(srn_file_tex)
# load yaml
ls_dict_yml = yaml.load(fl_yaml, Loader=yaml.BaseLoader)
# type
type(ls_dict_yml)
```

```
## <class 'list'>
```

```
type(ls_dict_yml[0])
# display
```

```
## <class 'dict'>
```

```
pprint.pprint(ls_dict_yml, width=1)
```

```
## [{'core': [{'code': "c('word1','word2')\n"
##                     'function()\n'
##                     'for '
##                     '(ctr '
##                     'in '
##                     'c(1,2)) '
##                     '{}\n',
##             'package': 'r'},
##            {'code': 'group_by()\n',
##             'package': 'dplyr'}],
##   'date': '2020-05-02',
##   'description': 'Frequency '
##                  'table, '
##                  'bar '
##                  'chart '
##                  'and '
##                  'histogram.\n'
```

```
##                  'R '
##                  'function '
##                  'and '
##                  'lapply '
##                  'to '
##                  'generate '
##                  'graphs/tables '
##                  'for '
##                  'different '
##                  'variables.\n',
##   'file': 'matrix_matlab',
##   'output': {'pdf_document': {'includes': {'in_header': '../preamble.tex'},
##                               'pandoc_args': '../_output_kniti_pdf.yaml'}},
##   'title': 'One '
##            'Variable '
##            'Graphs '
##            'and '
##            'Tables',
##   'urlcolor': 'blue'},
##  {'core': [{'code': 'setwd()\n',
##             'package': 'r'},
##            {'code': 'write_csv()\n',
##             'package': 'readr'}],
##   'date': '2020-05-02',
##   'date_start': '2018-12-01',
##   'description': 'Opening '
##                  'a '
##                  'Dataset.\n',
##   'file': 'matrix_algebra_rules',
##   'title': 'Opening '
##            'a '
##            'Dataset',
##   'titleshort': 'Opening '
##                 'a '
##                 'Dataset'},
##  {'description': 'Third '
##                  'file '
##                  'description.',
##   'file': 'matrix_two',
##   'title': 'Third '
##            'file',
##   'titleshort': 'Third '
##                 'file'}]
```

Select yaml information by *file* name which is a key shared by components of the list:

```
ls_str_file_ids = ['matrix_two']
ls_dict_selected = [dict_yml for dict_yml in ls_dict_yml if dict_yml['file'] in ls_str_file_ids]
pprint.pprint(ls_dc_selected, width=1)
```

```
## [{'date': datetime.date(2020, 5, 2),
##   'description': 'Frequency '
##                  'table, '
##                  'bar '
##                  'chart '
##                  'and '
##                  'histogram',
##   'file': 'mat_matlab',
##   'title': 'One '
```

```
##             'Variable '
##             'Graphs '
##             'and '
##             'Tables',
##    'val': 1}]
```

4.2.3.3 Dump List of Dictionary as YAML

- py yaml dump pipe

Given a list of dictionaries, dump values to yaml. Note that dumped output does not use pipe for long sentences, but use single quote and space line, which works with the rmdparrse.py function without problem.

```
ls_dict_selected = [dict_yml for dict_yml in ls_dict_yml
                    if dict_yml['file'] in ['matrix_two','matrix_matlab']]
print(yaml.dump(ls_dict_selected))
```

```
## - core:
##   - code: 'c(''word1'',''word2'')
##
##       function()
##
##       for (ctr in c(1,2)) {}
##
##       '
##     package: r
##   - code: 'group_by()
##
##       '
##     package: dplyr
##   date: '2020-05-02'
##   description: 'Frequency table, bar chart and histogram.
##
##     R function and lapply to generate graphs/tables for different variables.
##
##     '
##   file: matrix_matlab
##   output:
##     pdf_document:
##       includes:
##         in_header: ../preamble.tex
##       pandoc_args: ../_output_kniti_pdf.yaml
##   title: One Variable Graphs and Tables
##   urlcolor: blue
## - description: Third file description.
##   file: matrix_two
##   title: Third file
##   titleshort: Third file
```

4.3 Install Python

4.3.1 Core Installations

Go back to fan's Python Code Examples Repository (bookdown site).

Use the PyYAML to parse yaml.

4.3.1.1 Git Bash

1. Download and install git

4.3.1.2 Conda Install

1. Download Anaconda for Python 3. For more involved conda instructions see here
2. Get where you installed conda: open up *anaconda prompt* with admin rights (press windows button, and search for anaconda prompt, right click on the resulting terminal icon, choose as admin, a terminal opens up).

```
where python
where anaconda

# C:/ProgramData/Anaconda3/Scripts/anaconda.exe
# C:/ProgramData/Anaconda3/python.exe
```

3. Add to Path: open up windows *Path* and copy the paths found above inside.

4.3.1.2.1 Add To Path Details To Add Anaconda to Path, In Windows

1. Search for: Environment Variables
2. Edit Environment Variables
3. Add new to Path (lower half):
   - C:/ProgramData/Anaconda3/Scripts/
   - C:/ProgramData/Anaconda3/
4. Now open up regular windows command Prompt, Type in: conda –version
5. Close and Open up Git Bash: conda –version

Alternatively, in windows, directly search for Path, and add the python and anaconda exe paths to paths.

Appendix A

Index and Code Links

A.1 Array, Matrix, Dataframe links

A.1.1 Section 1.1 Array links

1. Python String Manipulation Examples: **rmd** | **r** | **pdf** | **html**
   - Search for substring, replace string, wrap string.
   - **py**: *zip() + upper()*
   - **textwrap**: *fill(st, width = 20)*

A.1.2 Section 1.2 Dictionary links

1. Python Dictionary Exampls and Usages: **rmd** | **r** | **pdf** | **html**
   - List comprehension with dictionary
   - **py**: *dc = {'key': "name", 'val': 1}*

A.2 Tables and Graphs links

A.2.1 Section 2.1 Matplotlib Base Plots links

1. Mabplotlib Scatter and Line Plots: **rmd** | **r** | **pdf** | **html**
   - Plot several arrays of data, grid, figure title, and line and point patterns and colors.
   - Plot out random walk and white noise first-order autoregressive processes.
   - **matplotlib**: *subplots() + ax.plot() + ax.legend() + ylabel() + xlabel() + title() + grid() + show()*
   - **numpy**: *random.normal() + random.seed() + cumsum() + arange()*
2. Mabplotlib Text Plots: **rmd** | **r** | **pdf** | **html**
   - Print text as figure.
   - **matplotlib**: *ax.text()*
   - **textwrap**: *fill()*
   - **json**: *dump()*

A.3 Get Data links

A.3.1 Section 3.1 Environmental Data links

1. CDS ECMWF Global Enviornmental Data Download: **rmd** | **r** | **pdf** | **html**
   - Using Python API get get ECMWF ERA5 data.
   - Dynamically modify a python API file, run python inside a Conda virtual environment with R-reticulate.
   - **r**: *file() + writeLines() + unzip() + list.files() + unlink()*
   - **r-reticulate**: *use_python() + Sys.setenv(RETICULATE_PYTHON = spth_conda_env)*

A.4 System and Support links

A.4.1 Section 4.1 Command Line links

1. Execute Python from Command Line and Run Command Line in Python: **rmd** | **r** | **pdf** | **html**
   - Run python functions from command line.
2. Run Matlab Command Line Operations: **rmd** | **r** | **pdf** | **html**
   - Generate a matlab script and run the script with parameters.
   - **subprocess**: *cmd = Popen(ls_str, stdin=PIPE, stdout=PIPE, stderr=PIPE) + cmd.communicate()*
   - **decode**: *decode('utf-8')*
   - **os**: *chdir() + getcdw()*

A.4.2 Section 4.2 File In and Out links

1. Python Reading and Writing to File Examples: **rmd** | **r** | **pdf** | **html**
   - Reading from file and replace strings in file.
   - Convert text file to latex using pandoc and clean.
   - Search for files in several folders with file substring.
   - Get path root, file name, file stem, etc from path.
   - **py**: *open() + write() + replace() + [c for b in [[1,2],[2,3]] for c in b]*
   - **subprocess**: *call()*
   - **pathlib**: *Path().rglob() + Path().stem*
   - **os**: *remove() + listdir() + path.isfile() + path.splitdrive() + os.path.splitext() + os.path.split()*
2. Python Directory and Folder Operations: **rmd** | **r** | **pdf** | **html**
   - Generate new folders and files.
   - Generate subfolder recursively.
   - Copying and moving files across folders.
   - Aggregate subfolders into a folder and move.
   - **py**: *open(srt, 'w') + write() + close()*
   - **os**: *os.listdir() + os.path.join('/', 'c:', 'fa', 'fb')*
   - **pathlib**: *Path(srt).mkdir(parents=True, exist_ok=True) + [Path(spn).stem for spn in Path(srt).rglob(st)]*
   - **shutil**: *shutil.copyfile('/fa/fl.txt', '/fb/fl.txt') + shutil.copy2('/fa/fl.txt', '/fb') + shutil.rmtree('/fb')*
   - **distutils**: *dir_util.copy_tree('/fa', '/fb')*
3. Python Yaml File Parsing: **rmd** | **r** | **pdf** | **html**
   - Parse and read yaml files.
   - **yaml**: *load(fl_yaml, Loader=yaml.BaseLoader) + dump()*
   - **pprint**: *pprint.pprint(ls_dict_yml, width=1)*

A.4.3 Section 4.3 Install Python links

1. Basic Conda Setup Instructions: **rmd** | **r** | **pdf** | **html**
   - Conda and git installations
   - **bash**: *where*

Bibliography

Xie, Y. (2020). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.18.