Running Picasso on the server (Internal tutorial)

Let's start from a server without any pre-installations.

Download

```
cd ~   # navigate to your home directory
git clone https://github.com/ZPYin/Pollynet_Processing_Chain

cd Pollynet_Processing_Chain
git fetch origin dev:dev   # fetch the dev branch for feature developments
```

Requirements

- **Anaconda3** to provide the Python interpreter and easy packages management
- **> MATLAB 2014a** (Only MATLAB 2014a and 2018b have been tested)
- MySQL connector

commands to install the dependencies

```
cd ~
mkdir tmp
cd tmp

# Anaconda3
wget https://repo.anaconda.com/archive/Anaconda3-2019.07-Linux-x86_64.sh
bash Anaconda3-2019.07-Linux-x86_64.sh
# follow the installation instructions

# install python dependencies
# Anaconda3 has built-in support of `matplotlib`, `numpy` and `scipy`
```

MySQL connector for MATLAB

```
cd ~/tmp

# download the mysql connector
wget https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-
5.1.48.tar.gz

# unzip mysql connector into '~/include'
mkdir ~/include
unzip mysql-connector-java-5.1.48.tar.gz -d ~/include
```

you can also check the MySQL website for details. Please choose the right **version** (only 5.1), otherwise you will fail to connect the to MySQL database.

```
# Add the mysql connector into MATLAB search path
# Create 'javaclasspath.txt' under ~/.matlab/{version}/
# with inserting an entry which specifies the absolute path to the .jar
file
# Restart matlab
```

Note: Be sure the **mysql connector** works before you continue the tutorial.

Configuration

Details about the configurations can be found here

Usage

There are several bash scripts to automate your work with using the **Pollynet_Processing_Chain** (Picasso), which were located under `../lib/script`.

In general, it consists of two (three) steps as below:

- Unzip Polly data into `todo_filelist` and write file info into `fileinfo_new.txt`
- Activate **Picasso** to process the polly data and prepare the `done_filelist.txt`
- Scan `done_filelist.txt` and add the image info into the Polly database (Picasso_go.sh)

Process the Polly data at the given day

```
# process the data from pollyxt_tjk at 2019-10-01
~/Pollynet_Processing_Chain/lib/script/pollynet_process_day -d 20191001 -p
pollyxt_tjk -f /pollyhome/pollyxt_tjk -c
pollynet_processing_chain_config.json
```

Instructions with using `pollynet_process_day.sh`

```
(base) [Picasso@rsd ~]$ pollynet_process_day -h
Usage:
/pollyhome/Picasso/Pollynet_Processing_Chain/lib/script/pollynet_process_d
ay.sh [option...] {today|yesterday}

Process the polly data at any give time.
   -d, --yyyymmdd          set the date for the polly data
   -p, --polly_type        set the instrument type
                           - PollyXT_LACROS
                           - PollyXT_TROPOS
                           - PollyXT_NOA
                           - PollyXT_FMI
```

```
                            - PollyXT_UW
                            - PollyXT_DWD
                            - PollyXT_TJK
                            - PollyXT_TAU
                            - arielle
                            - Polly_1v2
   -f, --polly_folder       specify the polly data folder
                            e.g., '/pollyhome/pollyxt_lacros'
   -c, --config_file        specify the pollynet processing file for the
data processing
                            e.g., 'pollynet_processing_chain_config.json'
   -h, --help               show help message
```

Process the Polly data between two given dates

```
# process the data from pollyxt_tjk between 2019-10-01 and 2019-10-05
~/Pollynet_Processing_Chain/lib/script/pollynet_process_history_data.sh -s
20191001 -e 20191005 -p pollyxt_tjk -f /pollyhome/pollyxt_tjk -c
pollynet_processing_chain_config.json
```

Instructions with using `pollynet_process_history_day.sh`

```
(base) [Picasso@rsd ~]$ pollynet_process_history_data -h
Usage:
/pollyhome/Picasso/Pollynet_Processing_Chain/lib/script/pollynet_process_h
istory_data.sh [option...]

Process the polly history data.
   -s, --start_date         set the start date for the polly data
                            e.g., 20110101
   -e, --end_date           set the end date for the polly data
                            e.g., 20150101
   -p, --polly_type         set the instrument type
                            - PollyXT_LACROS
                            - PollyXT_TROPOS
                            - PollyXT_NOA
                            - PollyXT_FMI
                            - PollyXT_UW
                            - PollyXT_DWD
                            - PollyXT_TJK
                            - PollyXT_TAU
                            - arielle
                            - Polly_1v2
   -f, --polly_folder       specify the polly data folder
                            e.g., '/pollyhome/pollyxt_lacros'
   -c, --config_file        specify the pollynet processing file for the
data processing
                            e.g., 'pollynet_processing_chain_config.json'
   -h, --help               show help message
```

Automate the processing for the entire PollyNET data

```
# Process the data during the past 7 days that was not loaded into the
database yet
# and scan the done_filelist.txt after all the results were output
~/Pollynet_Processing_Chain/lib/script/Picasso_go.sh -p
/pollyhome/Picasso/pollyAPP/config/config.private -c
pollynet_processing_chain_config.json --check_gdas true
```

Instructions with using `Picasso_go.sh`

```
(base) [Picasso@rsd ~]$ Picasso_go -h
Usage:
/pollyhome/Picasso/Pollynet_Processing_Chain/lib/script/Picasso_go.sh
[option...]

Start Picasso.
   -p, --pollyapp_config   specify the path of 'config.private' for the
pollyAPP
   -g, --check_gdas        flag to control whether to reprocess the data
when GDAS if ready (true | false)
   -c, --config_file       specify the pollynet processing file for the
data processing
                           e.g., 'pollynet_processing_chain_config.json'
   -h, --help              show help message
```

Schedual the script

Below is an example that was configured for our live environment.

```
# For those who work with the crontab
# Please keep your task well-documented, in case others can keep going.

# setup the environment for perl and Anaconda3
PATH=/bin:/usr/bin:/pollyhome/Picasso/.perlbrew/libs/perl-
5.22.2\@devel/bin:/pollyhome/Picasso/perl5/perlbrew/bin:/pollyhome/Picasso
/perl5/perlbrew/perls/perl-5.22.2/bin:/pollyhome/Picasso/anaconda3/bin
PERL5LIB=/pollyhome/Picasso/.perlbrew/libs/perl-5.22.2@devel/lib/perl5

# Picasso real-time analysis
0 2-23 * * * /usr/bin/flock -x -w 1200 /tmp/ms.cron2.lockfile -c
'/pollyhome/Picasso/Pollynet_Processing_Chain/lib/script/Picasso_go.sh >
/pollyhome/Picasso/log_push_current_pollydata 2>&1'

# Picasso reanalysis with GDAS1 data
1 0 * * * /usr/bin/flock -x -w 1200 /tmp/ms.cron2.lockfile -c
```

```
'/pollyhome/Picasso/Pollynet_Processing_Chain/lib/script/Picasso_go.sh -g
true > /pollyhome/Picasso/log_push_current_pollydata 2>&1'

# Update the GDAS1 station, AERONET and radiosonde station lsit
0 1 * * *
/pollyhome/Picasso/Pollynet_Processing_Chain/lib/script/update_gdas1_site_
list /pollyhome/Picasso/Pollynet_Procssing_Chain/doc/gdas1-site-list.txt
2 1 * * *
/pollyhome/Picasso/Pollynet_Processing_Chain/lib/script/update_radiosonde_
and_aeronet_site_list.sh

# scan the trajectory results into the database
# 10 1 * * * source activate traj_scanner; python
/pollyhome/Picasso/trajectory_results/traj_file_scanner/src/trajectory_sca
nner.py; add_new_data2pollydb
/pollyhome/Picasso/done_filelist/done_filelist_trajectory.txt
```