Precipitable-water Model Analysis Tool Official Manual

Spencer Riley

Jul 22, 2022

CONTENTS:

The Precipitable-Water Model Analysis Tool is an open-source suite for analyzing the relationship between atmospheric brightness temperature and precipitable water.

Warning: This documentation is under active development.

CHAPTER

ONE

GETTING STARTED

1.1 Introduction

The Precipitable-water Model Analysis Tool (`PMAT`) is a computational utility that is used to analyze the data collected from this project to understand the relationship between the zenith sky temperature and precipitable water in the atmosphere. `PMAT` has three different modules that work together to present data.

The first is the Deployment Module. This module acts as the user interface for the software suite, whether it be locally or through cloud services.

The second is the Pre-processing Module, this module imports data from University of Utah's MesoWest and the University of Wyoming UpperAir Databases.

The third module is the main program to run the analysis, the DAnalysis Module. Here the all of the data is presented and the regression analysis between precipitable water and zenith sky temperature is conducted.

1.2 Installation and Deployment Tutorial

We also require two data files. One that contains the raw data collected by the temperature sensors, that also includes date and time information (`cool_data.csv`). The second should contain sensor definitions with additional parameters for the preprocessing and analysis phases (`_pmat.yml`). A template and detailed breakdown of the configuration file is provided in Chapter 2.1, followed by a detailed breakdown on the data file format.

1.2.1 Github

This version of the Deployment module is, for the most part, automated and recommended. Follow the steps in this section to successfully deploy PMAT through GitHub with GitHub Actions.

1. Create a GitHub repository from the template repository[1].
2. Edit the README.md page based on your location and username
3. Update all files that are contained in the `data/` directory, and utilize the documentation on data formatting that is provided
4. Upon finializing updates on `cool_data.csv`, the workflow will run automatically and the visual and data products will be generated

[1] https://template.pmat.app

1.2.2 Amazon Web Service (AWS)

For Amazon Web Services, `PMAT` can be configured through the EC2 virtual machines. Once they have been configured, connect to the virutal machine. Once connected, enter the following commands

```
sudo yum update -y
```

```
sudo amazon-linux-extras install docker
```

```
sudo docker pull ghcr.io/physicsgoddess1972/pmat:latest
```

From here, data files can be added and utilizing the `local_deploy.sh` script, `PMAT` can be executed.

1.2.3 Google Cloud Console (GCloud)

1.2.4 Local

We fully support Ubuntu and Debian systems. We do have minimal Windows support through the usage of Windows Subsystem for Linux (WSL) and Virutalization.

1.2.5 Development

CHAPTER

TWO

WORKING WITH DATA AND PMAT

This chapter will discuss the data components associated with PMAT. The first section will detail the formatting guidelines of the input files, followed by another discussion regarding the various output files generated by the software suite.

2.1 Input Data Formatting

The two input files discussed in this section include a YAML configuration file and a Comma Separated Value data file.

2.1.1 Configuration Input

The role of the configuration file is to store a series of parameters that includes sensor information, analytic parameters, logging options, and the site identifiers for the University of Wyoming's Upper-Air database and the University of Utah's MesoWest database.

See also:

Chapter 5.1.1 shows the structure of the data fields. The filename of this file must be `_pmat.yml`.

Sensor fields

sensor.name ***(string)*****:**

Detail
The name of the sensor.

Note
If there are multiple of the same sensor use the notation `_N` with N being the index of the sensor.

Example
Sensor 09, Sensor 10_1, Sensor 10_2

sensor.error ***(float)*****:**

OPTIONAL
Detail
The manufacturer reported error on the sensor

Example
2.5, 5.0

sensor.color ***(string)*****:**

> **Detail**
> A hexadecimal color code that will be used to identify the sensor on visualizations.
>
> **Example**
> FF0000, 0000FF

sensor.ratio ***(string)*****:**

> **OPTIONAL**
> **Detail**
> The distance to spot ratio that is reported by the manufacturer.
>
> **Example**
> 12 to 1, 21 to 1

sensor.emissivity ***(float)*****:**

> **OPTIONAL**
> **Detail**
> The emissivity of the sensor as reported by the manufacturer.
>
> **Example**
> 0.95

sensor.poster ***(boolean)*****:**

> **Detail**
> A boolean that will decide whether the sensor will be shown in the poster-specific plots
>
> **Example**
> true, false

sensor.active ***(boolean)*****:**

> **Detail**
> A boolean that will decide whether the sensor will be used in the analysis.
>
> **Example**
> true, false

Analysis

train_fraction ***(float)*****:**

> **Detail**
> The fraction of data being used to create the training set.
>
> **Note**
> A value between 0 and 1.
>
> **Example**
> 0.8

rel_difference (float):

Detail

Example
2

iteration.step ***(integer)*****:**

Detail
The number of steps the analysis will run

Example
1, 100, 1000

Logging

verbose ***(string)*****:**

Detail
An identifier for the level of logging

Example
DEBUG, WARN, ERROR, INFO

Import

For information regarding the usage of external files for PWV or RH measurements, refer to …

mesowest.id ***(string)*****:**

Detail
The measurement site identifier for the MesoWest database

Example
KONM, KRAP

wyoming.id ***(string)*****:**

Detail
The measurement site identifier for the Wyoming Upper-Air database

Example
ABQ, EPZ

wyoming.weight ***(string)*****:**

Detail
The weighting used on the PWV measurements for analysis.

Note
If there is multiple sites, these values should add to 0.5.

Example
0.4, 0.2, 0.5

2.1.2 Raw Data File

The raw data file is processed, through pattern identification, allowing for a flexible format with few strict requirements. One of these requirements is that the sky and ground temperature should be separated into groups and ordered the same way as the configuration file. Here are three examples of data files:

- Dataset Example 1[2]
- Dataset Example 2[3]
- Dataset Example 3[4]

It should be noted that the columns do not have to be in any set order, with one small caveat, the model pulls the data from columns with headers containing specific words or phrases. The caveat is with regards to Ground and Sky temperature readings. The temperature measurements must go in consecutive order by sensor as determined by `_pmat.yml`.

For example, if the order of the sensors in `_pmat.yml` is 1610 TE, FLIR i3, and then AMES 1. Then the order of the ground and sky temperature measurements in the dataset should be: 1610 TE, FLIR i3, and then AMES 1. (As seen in Dataset 2).

Date ***(datetime, ``YYYY-MM-DD``)*:**

> **Detail**
> The date of the measurements.

Time ***(datetime, ``HH:MM``)*:**

> **Detail**
> The local time of the measurements

Sky temperature ***(float)*:**

> **Detail**
> The sky temperature measurements. The header of this column should be Sensor Name (Sky), where Sensor Name is the name of the sensor used in the configuration file.

Ground temperature ***(float)*:**

> **Detail**
> The ground temperature measurements. The header of this column should be Sensor Name (Ground), where Sensor Name is the name of the sensor in the configuration file.

[2] https://github.com/physicsgoddess1972/Precipitable-Water-Model/blob/master/data/example/example1.csv
[3] https://github.com/physicsgoddess1972/Precipitable-Water-Model/blob/master/data/example/example2.csv
[4] https://github.com/physicsgoddess1972/Precipitable-Water-Model/blob/master/data/example/example1.csv

2.2 Output Data Formatting

There are a variety of data files generated by the software suite. The data files are stored as CSV files, with each row presenting data for a single day.

2.2.1 General data files

The primary data file [`master_data.csv`] generated is the full dataset that includes:

- Date
- time
- sky condition (clear sky/overcast)
- ground temperature
- sky temperature
- Radiosonde PWV
- Relative Humidity
- Dewpoint
- User comments

2.2.2 Machine learning

The machine learning data file includes five columns:

- Date
- Average brightness temperature
- Average PWV
- Relative Humidity
- Sky Condition

This data set supports the classification of data by the sky condition label.

2.2.3 Analytic results

The main analytical results are stored as YAML configuration files. The results of each step in the iterative analysis process are saved to a file with the name `_output.yml`. An example of this file is presented below. [sample of _output.yml] [table of the fields in _output.yml]

The averaged results of the steps are also stored in a YAML file. [sample of _results.yml] [table of the fields in _results.yml]

CHAPTER

THREE

REPORTING ISSUES WITH PMAT

3.1 Bug reporting

The developers of this project are human and can make mistakes.

3.1.1 Opening a Ticket

3.2 Feature requests

CHAPTER

FOUR

CHANGELOG

4.1 PMAT Cirrus

Version
2.0

Date
6 Mar 2021

Tagline
New and Improved PMAT

4.1.1 Overall

- [Updated] Compatible with R 4.0

4.1.2 Data-input

- [Added] Now includes relative humidity imports.
- [Added] Now pulls data from MesoWest..
- [Added] New guidelines for sensors that are not active (See Documentation Page for further info.)

4.1.3 Setup-script

- [Updated] Now installs R 4.0
- [Added] Additional argument to configure database imports (run *bash setup.sh -h* for more information)

4.1.4 Plots

- [Fixed] Fixed issues with bar charts where if there were more than three sensors, not all bar charts would be added for the remaining sensors.
- [Added] Added more time series plots and more composite plots.
- [Updated] Changed the x-axis labeling system to have tick marks at the 1st of the month.
- [Updated] Redesigned the main analytical plot, confidence interval is now a shaded region, and the plot is now monochromatic.
- [Updated] Pac-Man residual was removed from this plot set.
- [Updated] Pac-man residual now resides in a new plot set (run *Rscript model.r –pacman*)
- [Added] Mean TPW and Mean temperature comparison can now be visualized in a Pac-Man plot.

4.1.5 Web-applications

- [Added] Two web-apps are active. One is a Data Dashboard, which allows for the viewing of time series data as a scatter plot or a heat map, and analytical comparisons between data that has been collected.
- [Added] The Data Dashboard also allows for custom time series data to be uploaded.
- [Added] The Machine Learning dashboard now allows for custom data to be uploaded.

4.1.6 Documentation

- [Fixed] Fixed multiple CSS issues.
- [Updated] Altered Pac-Man residual plot documentation to refer to the package documentation
- [Updated] Updated procedure to include the new command-line arguments
- [Added] Included buttons on the dashboard's "Project Updates" card to include Pac-Man plots and Poster plots that are generated from data we have collected.
- [Updated] We also scored a *.tech* domain for the page.

4.1.7 Automation

- [Misc] This is a work in progress

4.2 PMAT Altocumulus

Version
1.0

Date
10 Nov 2019

Tagline
Initial Deployment of The Precipitable Water Model

4.2.1 Overall

- [Added] Flexible data input
- [Added] Easy Hands-off setup.
- [Added] Command-line arguments to access the different plots available
- [Added] Time Series plots for zenith sky temperature and precipitable water
- [Added] Analytical plots showing the correlation between zenith sky temperature and precipitable water
- [Added] Poster ready plots for presentations
- [Added] A data set including the average temperature and precipitable water
- [Added] The Pac-Man Residual
- [Updated] Documentation Page.

CHAPTER

FIVE

TEMPLATES

5.1 Input Data File

5.1.1 _pmat.yml

```
- instruments:
  - sensor:
      name:
      error:
      color:
      ratio:
      range:
      emissivity:
      poster:
      active:
- analysis:
  - train_fraction:
  - rel_difference:
  - iteration:
      step:
      seed:
- logging:
  - verbose:
- import:
  - mesowest:
    - id:
  - wyoming:
    - id:
      weight:
```

CHAPTER

SIX

PMAT REFERENCE

6.1 pmat_analysis.r

module
Precipitable Water Model Analysis Tool: Analysis

synopsis
This module contains analysis functions

`exp.`**`regression`**(*t=NULL*, *mean.out*)

Detail
Function includes all of the stuff to generate the exponential regression model with intervals

Parameters

- **t** (*double*) – training fraction
- **mean.out** (*list*) – the output of mean.filter

Returns
returns the data series and model statistics

Return type
list

`lin.`**`regression`**(*x*, *y*)

Detail
Linear regression function

Parameters

- **x** (*double*) – the domain of the dataset
- **y** (*double*) – the range of the dataset

Returns
returns the data series and model statistics

Return type
list

`data.`**`partition`**(*x*, *y*, *tr.sz=tr.sz*)

Detail
splits the data into a training/testing set

Parameters

- **x** (*double*) – domain of the data
- **y** (*double*) – range of the data
- **tr.sz** (*double*) – fraction of the data in the testing set

Returns
a list containing the training and testing sets

Return type
list

`iterative.`**`analysis`**`(`*out.bool*, *mean.out*`)`

Detail
computes regression statistics and outputs to a yaml file

Parameters

- **out.bool** (*logical*) – determine whether to generate new _output.yml
- **mean.out** (*list*) – output of mean.filter

Returns
iterative stats and _output.yml

Return type
list

`lsvm``(`*x*, *y*, *l*, *tr.sz=0.7*, *seed=sample(1:2^15, 1)*`)`

Detail
Generates a Linear Support Vector Machine and draws the decision hyperplane and support vectors

Parameters

- **x** (*double*) – domain of dataset
- **y** (*double*) – range of dataset
- **l** (*double*) – labels of the dataset
- **tr.sz** (*double*) – fraction of data to be used for model training
- **seed** (*integer*) – the random seed

Returns
list of data, labels, and the coefficients

Return type
list

6.2 pmat_processing.r

module
Precipitable Water Model Analysis Tool: Pre-processing

synopsis
functions for preprocessing

`colscheme`(*range*, *set='Set1'*)

Detail
a function that generates an array of colors based on the number of elements

Parameters
range (*list*) – a list of data series

Returns
a list of colors

Return type
list

`nan.filter`(*stuff*)

Detail
removes nan values from a set of lists

Parameters
stuff (*list*) – list of arrays

Returns
returns list with filtered data and the indices with nans

Return type
list

`mean.filter`(*nan.out*, *n*)

Detail
filters the data based on the comparison of the daily std and the average std of the dataset

Parameters

- **nan.out** (*list*) – the output of nan.filter
- **n** (*integer*) – threshold

Returns
an array of indices for PWV values to be analyzed

Return type
list

`dna.filter`(*filter.res*)

Detail
removes data labels as Do Not Analyze

Parameters
`filter.res` (*list*) – overcast.filter results

Returns
overcast.filter results with DNA points removed

Return type
list

`inf.counter`(*bool*, *snsr_data*, *label*)

Detail
identifies the -Inf values

Parameters

- `bool` (*logical*) – decides if -Inf is not replaced with NaN
- `snsr_data` (*list*) – the dataset
- `label` (*character*) – the identifier for the dataset (e.g. sky, gro, skyo, groo)

Returns
data set that replaces all -Infs for NaN (If bool == FALSE).

Return type
list

`index.norm`(*x*)

Detail
calculates the normalized index of the dataset

Parameters
`x` (*double*) – data range

Returns
an array of values between 0 and 1

Return type
double

`overcast.filter`(*col_con*, *col_date*, *col_com*, *pw_name*, *snsr_name*, *cloud.bool*)

Detail
Filters our data with overcast condition

Parameters

- `col_con` (*integer*) – column index for condition labels
- `col_date` (*integer*) – column index for date stamp
- `col_com` (*integer*) – column index for comments
- `pw_name` (*list*) – pw measurement labels
- `snsr_name` (*list*) – sensor labels
- `cloud_bool` (*logical*) –

Returns
A list of lists containing either clear-sky/overcast data

Return type
list

`sky.processing`(*filter.res*)

Detail
Computes average values and weighted averages

Parameters
`filter.res` (*`list`*) – results of the overcast.filter function

Returns
series of arrays including average PWV, RH, etc.

Return type
list

6.3 pmat_products.r

module
Precipitable Water Model Analysis Tool: Products

synopsis
plotting functions for PMAT

`time.pwindex`(*datetime*)

Detail
Normalized PWV index for both clear sky and overcast data

Parameters
`datetime` – the datestamp of the data

`time.nth_range`(*y*, *title*, *color*, *leg.lab*, *ylab*, *datetime*, *over.bool*)

Detail
Multirange Time Series plot series

Parameters

- **`y`** (*`list`*) – the range of the plot
- **`title`** (*`character`*) – the title/description of the plot
- **`color`** (*`character`*) – the color string for the data range
- **`leg.lab`** (*`list`*) – a list of
- **`ylab`** (*`character`*) – the y-axis label
- **`datetime`** (*`double`*) – the datetime array
- **`over.bool`** (*`logical`*) – the condition of data (clear sky/overcast)

`time.composite`(*y*, *title*, *color*, *ylab*, *datetime*, *over.bool*)

Detail
Time Series composite plot series

Parameters

- **`y`** (*`list`*) – the range of the plot
- **`title`** (*`character`*) – the title/description of the plot
- **`color`** (*`character`*) – the color string for the data range
- **`ylab`** (*`character`*) – the y-axis label
- **`datetime`** (*`double`*) – the datetime array
- **`over.bool`** (*`logical`*) – the condition of data (clear sky/overcast)

`time.`**`mono_composite`**`(`*y*, *title*, *ylab*, *datetime*, *over.bool*`)`

Detail
Time Series composite plot series

Parameters

- **`y`** (*`list`*) – the range of the plot
- **`title`** (*`character`*) – the title/description of the plot
- **`ylab`** (*`character`*) – the y-axis label
- **`datetime`** (*`double`*) – the datetime array
- **`overcast`** (*`logical`*) – the condition of data (clear sky/overcast)

`time.`**`multiyear`**`(`*y*, *title*, *color*, *datetime*, *ylab*, *over.bool*`)`

Detail
Climatology plot

Parameters

- **`y`** (*`list`*) – the range of the plot
- **`title`** (*`character`*) – the title/description of the plot
- **`color`** (*`character`*) – the color string for the data range
- **`datetime`** (*`double`*) – the datetime array
- **`ylab`** (*`character`*) – the y-axis label
- **`overcast`** (*`logical`*) – the condition of data (clear sky/overcast)

`analysis.`**`nth_range`**`(`*over.bool*, *x*, *y*, *title*, *label*, *color*, *leg.lab*`)`

Detail
Comparative analysis plot

Parameters

- **`overcast`** (*`logical`*) – the condition of data (clear sky/overcast)
- **`x`** – the domain of the plot
- **`y`** – the range of the plot
- **`title`** – the title/description of the plot
- **`label`** –
- **`color`** – the color string for the data

- **leg.lab** –

`analysis.`**`regression`**(*over.bool*, *x*, *y*, *title*, *label*, *iter*)

Detail
Super Average Plot with Exponential Fit

Parameters

- **overcast** (*bool*) – the condition of data (clear sky/overcast)
- **x** (*double*) – the domain of the plot
- **y** (*double*) – the range of the plot
- **title** (*character*) – the title/description of the plot
- **label** (*list*) – list containing the x-axis and y-axis labels
- **iter** (*list*) – the output of iter.analysis

`analysis.`**`svm`**(*model*)

Detail
plots SVM

Parameters
model (*list*) – output of lsvm

`pac.`**`compare`**(*over.bool*, *title*, *x*, *y*, *angular*, *radial*)

Detail
Pac-Man plot of Super Average Plot

Parameters

- **overcast** (*logical*) – the condition of data (clear sky/overcast)
- **title** (*character*) – the title/description of the plot
- **x** (*double*) – the domain of the plot
- **y** (*double*) – the range of the plot
- **angular** (*character*) – the angular axis label and units
- **radial** (*character*) – the radial axis label and units

`pac.`**`regression`**(*over.bool*)

Detail
Pac-Man residual plot

Parameters
over.bool (*bool*) – the condition of data (clear sky/overcast)

`chart.`**`histogram`**(*range*, *xlabel*, *title*)

Detail
Histograms of defined quantities

Parameters

- **range** (*double*) – the range of the plot

- **xlabel** (*list*) – the x-axis label of the plot
- **title** (*list*) – the title/description of the histogram

`poster.`**`plots`**(*over.bool*, *iter*, *mean.out*)

Detail
The set of all poster

Parameters

- **overcast** (*bool*) – the condition of data (clear sky/overcast)
- **iter** (*list*) – output of iterative.analysis
- **mean.out** (*list*) –

Returns
All available poster plots

`poster1`(...)

`poster2`(*over.bool*, *iter*, *mean.out*)

Detail
The analytics poster plot

Parameters

- **overcast** (*logical*) – the condition of data (clear sky/overcast)
- **iter** – the output of iterative.analysis
- **mean.out** –

`sensor.`**`chart`**(...)

Detail
sensor specific data type distribution charts

`sensor.`**`time`**(*over.bool*)

Detail
Instrumentation time series plots

Parameters
over.bool (*logical*) – the condition of data (clear sky/overcast)

`data.`**`gen`**(*over.bool*, *dir*)

Detail
creates a datafile containing the date, avg temp, and avg pwv for a defined condition

Parameters

- **over.bool** (*bool*) – the condition of the data (clear sky/overcast)
- **dir** – directory path

`data.`**`ml`**(*out.dir*)

Detail
creates a datafile containing the machine learning relavant information

Parameters
`out.dir` (*character*) – directory path

`data.step`(*seed*, *i*, *coef*, *r*, *S*)

Detail
writes a yaml object that contains the analysis results of each step in the iterative analysis

Parameters

- **seed** (*integer*) – the generated seed
- **i** (*integer*) – the step number
- **coef** (*list*) – the coefficients of the best-fit
- **r** (*double*) – the RSME value
- **S** (*double*) – the standard deviation

Returns
a yaml object

Return type
list

`data.final`(*out.dir*, *lengths*, *frac.kept*, *coef*, *std*, *rmse*, *over.bool=args$overcast*)

Detail
writes the final results of iterative.analysi

Parameters

- **out.dir** (*character*) – the output directory
- **lengths** (*list*) – list of lengths
- **coef** (*list*) – the average coefficients of the best-fit
- **std** (*double*) – the average standard deviation
- **rmse** (*double*) – the average rsme values
- **over.bool** (*logical*) – the condition of data (clear sky/overcast)

`visual.products`(*set*, *mean.out*, *datetime=datetime*, *over.bool=args$overcast*)

Detail
saves plot sets

Parameters

- **set** (*character*) – the –set parameter
- **mean.out** – output of mean.filter
- **datetime** – the datetime range of the data
- **over.bool** (*logical*) – the condition of data (clear sky/overcast)

Module
Precipitable Water Model Analysis Tool: Products

Synopsis
plotting functions for PMAT

`time.`**`pwindex`**`(`*datetime*`)`

Detail
Normalized PWV index for both clear sky and overcast data

Parameters
`datetime` – the datestamp of the data

`time.`**`nth_range`**`(`*y*, *title*, *color*, *leg.lab*, *ylab*, *datetime*, *over.bool*`)`

Detail
Multirange Time Series plot series

Parameters

- **`y`** (*`list`*) – the range of the plot
- **`title`** (*`character`*) – the title/description of the plot
- **`color`** (*`character`*) – the color string for the data range
- **`leg.lab`** (*`list`*) – a list of
- **`ylab`** (*`character`*) – the y-axis label
- **`datetime`** (*`double`*) – the datetime array
- **`over.bool`** (*`logical`*) – the condition of data (clear sky/overcast)

`time.`**`composite`**`(`*y*, *title*, *color*, *ylab*, *datetime*, *over.bool*`)`

Detail
Time Series composite plot series

Parameters

- **`y`** (*`list`*) – the range of the plot
- **`title`** (*`character`*) – the title/description of the plot
- **`color`** (*`character`*) – the color string for the data range
- **`ylab`** (*`character`*) – the y-axis label
- **`datetime`** (*`double`*) – the datetime array
- **`over.bool`** (*`logical`*) – the condition of data (clear sky/overcast)

`time.`**`mono_composite`**`(`*y*, *title*, *ylab*, *datetime*, *over.bool*`)`

Detail
Time Series composite plot series

Parameters

- **`y`** (*`list`*) – the range of the plot
- **`title`** (*`character`*) – the title/description of the plot
- **`ylab`** (*`character`*) – the y-axis label
- **`datetime`** (*`double`*) – the datetime array

- **overcast** (*logical*) – the condition of data (clear sky/overcast)

`time.`**`multiyear`**(*y*, *title*, *color*, *datetime*, *ylab*, *over.bool*)

Detail
Climatology plot

Parameters

- **y** (*list*) – the range of the plot
- **title** (*character*) – the title/description of the plot
- **color** (*character*) – the color string for the data range
- **datetime** (*double*) – the datetime array
- **ylab** (*character*) – the y-axis label
- **overcast** (*logical*) – the condition of data (clear sky/overcast)

`analysis.`**`nth_range`**(*over.bool*, *x*, *y*, *title*, *label*, *color*, *leg.lab*)

Detail
Comparative analysis plot

Parameters

- **overcast** (*logical*) – the condition of data (clear sky/overcast)
- **x** – the domain of the plot
- **y** – the range of the plot
- **title** – the title/description of the plot
- **label** –
- **color** – the color string for the data
- **leg.lab** –

`analysis.`**`regression`**(*over.bool*, *x*, *y*, *title*, *label*, *iter*)

Detail
Super Average Plot with Exponential Fit

Parameters

- **overcast** (*bool*) – the condition of data (clear sky/overcast)
- **x** (*double*) – the domain of the plot
- **y** (*double*) – the range of the plot
- **title** (*character*) – the title/description of the plot
- **label** (*list*) – list containing the x-axis and y-axis labels
- **iter** (*list*) – the output of iter.analysis

`analysis.`**`svm`**(*model*)

Detail
plots SVM

Parameters
`model` (*list*) – output of lsvm

`pac.compare`(*over.bool*, *title*, *x*, *y*, *angular*, *radial*)

Detail
Pac-Man plot of Super Average Plot

Parameters

- **overcast** (*logical*) – the condition of data (clear sky/overcast)
- **title** (*character*) – the title/description of the plot
- **x** (*double*) – the domain of the plot
- **y** (*double*) – the range of the plot
- **angular** (*character*) – the angular axis label and units
- **radial** (*character*) – the radial axis label and units

`pac.regression`(*over.bool*)

Detail
Pac-Man residual plot

Parameters
over.bool (*bool*) – the condition of data (clear sky/overcast)

`chart.histogram`(*range*, *xlabel*, *title*)

Detail
Histograms of defined quantities

Parameters

- **range** (*double*) – the range of the plot
- **xlabel** (*list*) – the x-axis label of the plot
- **title** (*list*) – the title/description of the histogram

`poster.plots`(*over.bool*, *iter*, *mean.out*)

Detail
The set of all poster

Parameters

- **overcast** (*bool*) – the condition of data (clear sky/overcast)
- **iter** (*list*) – output of iterative.analysis
- **mean.out** (*list*) –

Returns
All available poster plots

`poster1`(...)

`poster2`(*over.bool*, *iter*, *mean.out*)

Detail
The analytics poster plot

Parameters

- **`overcast`** (*`logical`*) – the condition of data (clear sky/overcast)
- **`iter`** – the output of iterative.analysis
- **`mean.out`** –

`sensor.chart`(...)

Detail
sensor specific data type distribution charts

`sensor.time`(*over.bool*)

Detail
Instrumentation time series plots

Parameters
`over.bool` (*`logical`*) – the condition of data (clear sky/overcast)

`data.gen`(*over.bool*, *dir*)

Detail
creates a datafile containing the date, avg temp, and avg pwv for a defined condition

Parameters

- **`over.bool`** (*`bool`*) – the condition of the data (clear sky/overcast)
- **`dir`** – directory path

`data.ml`(*out.dir*)

Detail
creates a datafile containing the machine learning relavant information

Parameters
`out.dir` (*`character`*) – directory path

`data.step`(*seed*, *i*, *coef*, *r*, *S*)

Detail
writes a yaml object that contains the analysis results of each step in the iterative analysis

Parameters

- **`seed`** (*`integer`*) – the generated seed
- **`i`** (*`integer`*) – the step number
- **`coef`** (*`list`*) – the coefficients of the best-fit
- **`r`** (*`double`*) – the RSME value
- **`S`** (*`double`*) – the standard deviation

Returns
a yaml object

Return type
list

`data.`**`final`**`(`*out.dir*, *lengths*, *frac.kept*, *coef*, *std*, *rmse*, *over.bool=args$overcast*`)`

Detail
writes the final results of iterative.analysi

Parameters

- **`out.dir`** (*`character`*) – the output directory
- **`lengths`** (*`list`*) – list of lengths
- **`coef`** (*`list`*) – the average coefficients of the best-fit
- **`std`** (*`double`*) – the average standard deviation
- **`rmse`** (*`double`*) – the average rsme values
- **`over.bool`** (*`logical`*) – the condition of data (clear sky/overcast)

`visual.`**`products`**`(`*set*, *mean.out*, *datetime=datetime*, *over.bool=args$overcast*`)`

Detail
saves plot sets

Parameters

- **`set`** (*`character`*) – the –set parameter
- **`mean.out`** – output of mean.filter
- **`datetime`** – the datetime range of the data
- **`over.bool`** (*`logical`*) – the condition of data (clear sky/overcast)

6.4 pmat_run.r

module
Precipitable-Water Model Analysis Tool

synopsis
The main file for PMAT. Documentation available at <https://docs.pmat.app>.

module
Precipitable-Water Model Analysis Tool

synopsis
The main file for PMAT. Documentation available at <https://docs.pmat.app>.

6.5 pmat_utility.r

module
Precipitable Water Model Analysis Tool: Utility

synopsis
general functions for PMAT

`logg`(*msglevel*, *msg*, *dir=out.dir*, *lev='INFO'*)

Detail
creates log entries for _log.txt

Parameters

- **`msglevel`** (*`character`*) – log level of the message
- **`msg`** (*`character`*) – the message itself
- **`dir`** (*`character`*) – directory that stores the log file
- **`lev`** (*`character`*) – the log level of the user

`aloha.first()`

Detail
shows first time user information

`aloha.startup()`

Detail
shows title banner for program

`aloha.closing()`

Detail
cleans up files and ends the program

`reset_time`(*datetime*)

Detail
A function that sets the time to 00:00:00

Parameters
`datetime` (*`character`*) – a Date or datetime object

Returns
A datetime object with time 00:00:00

Return type
double

`time_axis_init`(*date*)

Detail
A function that calculates the min, max, and position of the tick marks for

Parameters
`date` (*`double`*) – A date or datetime object

Returns
The max, min, and tick mark positions

Return type
list

`time_axis`(*datetime*)

Detail
A function that sets the x-axis format for time series plots

Parameters
`datetime` (*`double`*) – A date or datetime object

`stnd_title`(*des*, *overcast*)

Detail
A function that generates the title based on

Parameters

- **`des`** (*`character`*) – the description of the plot
- **`overcast`** (*`logical`*) – the sky condition

Returns
a title string

Return type
character