**OBSALL Apps**

**(Observation Application)**

***Documentation (ver. 1)***

***Climate DT Team***

***April 2024***

**CONTENTS**

[**CHAPTER 1: INTRODUCTION** 4](#_Toc165283376)

[**1.1.** **Overview of OBSALL** 4](#_Toc165283377)

[**1.2.** **Purpose** 4](#_Toc165283378)

[**1.3.** **Key Features** 4](#_Toc165283379)

[**1.4.** **Contributing** 4](#_Toc165283380)

[**CHAPTER 2: INSTALLATION** 4](#_Toc165283381)

[**2.1.** **Prerequisites** 4](#_Toc165283382)

[**2.2.** **Installation on Lumi HPC** 5](#_Toc165283383)

[***2.2.1. Cloning/ Copying*** 5](#_Toc165283384)

[***2.2.2. Setting environment*** 5](#_Toc165283385)

[***2.2.3. Executing application*** 6](#_Toc165283386)

[***2.2.4. Requesting modelled data*** 7](#_Toc165283387)

[**CHAPTER 3: MAIN COMPONENTS/ PARTS OF OBSALL** 7](#_Toc165283388)

[**3.1.** **Part SYNOP (ground-based/surface observations)** 7](#_Toc165283389)

[**3.2.** **Monitoring for SYNOP Part** 8](#_Toc165283390)

[***3.2.1. Rank histograms for all synop stations for 00, 06, 12, and 18 UTCs*** 8](#_Toc165283391)

[***3.2.2. Standard plots for each synop station*** 9](#_Toc165283392)

[***3.2.3. Summary rank histograms for all synop stations*** 11](#_Toc165283393)

[**3.3.** **Part TEMP (radiosounding observations)** 14](#_Toc165283394)

[**3.4.** **Monitoring for TEMP Part** 15](#_Toc165283395)

[***3.4.1. Rank histograms for all radiosounding stations for 00 and 12 UTCs*** 16](#_Toc165283396)

[***3.4.2. Standard plots for each radiosounding station*** 17](#_Toc165283397)

[***3.4.3. Summary rank histograms for all radiosounding stations*** 18](#_Toc165283398)

[**3.5.** **Part AMSUA-A (satellite observations)** 21](#_Toc165283399)

[**3.6.** **Monitoring for AMSU-A Part** 22](#_Toc165283400)

[***3.6.1. Rank histograms for all areas for daily mean*** 23](#_Toc165283401)

[***3.6.2. Standard plots for each area*** 24](#_Toc165283402)

[***3.6.3. Summary rank histograms for all areas*** 26](#_Toc165283403)

[**CHAPTER 4: CONTRIBUTING TO OBSALL** 29](#_Toc165283404)

[**4.1.** **Reporting bugs** 29](#_Toc165283405)

[**4.2.** **Suggesting features** 29](#_Toc165283406)

[**4.3.** **Contributing to code** 29](#_Toc165283407)

[**4.4.** **Contributing to documentation** 29](#_Toc165283408)

[**CHAPTER 5: TROUBLESHOOTING AND FAQ** 30](#_Toc165283409)

[**CHAPTER 6: REFERENCES AND ACKNOWLEDGMENTS** 30](#_Toc165283410)

[**6.1.** **Citing OBSALL** 30](#_Toc165283411)

[**6.2.** **Acknowledgments** 30](#_Toc165283412)

[**6.3.** **Especial thanks** 30](#_Toc165283413)

# **CHAPTER 1: INTRODUCTION**

## **Overview of OBSALL**

The OBSALL Apps is the Observation Application. It has been developed for evaluation of climate models, known as Digital Twins of the Earth. It is processing and analyzing large volumes of observation and modelled data. It has 3 main components (or Parts) such as SYNOP (for ground-based/ surface observations), TEMP/ RADSOUND (for upper are sounding observations), and AMSU-A/ SATELLITE (satellite-based observations). Important note, observational data with only free and unrestricted license have been collected and utilized for development, testing, and implementation purposes in Observation DataBase (ODB) format.

## **Purpose**

The purpose of the OBSALL Apps is (i) to pre-process modeled data from DestinE climate model experiments; (ii) to extract observations (from ODB) - from synoptic (ground-based/ surface observations) and radiosounding (upper air sounding) stations, and satellites; (iii) to extract modeled data at observation times and locations, applying appropriate observation operator, and ODB added with model simulation values; and (iv) to calculate/ produce relevant statistics for monitoring including illustrative material.

Important note, the end result is the augmented ODB, which can be used for: (i) on-line monitoring of the simulation in near-real time, and (ii) posterior analysis of the ClimateDT models quality.

## **Key Features**

* Pre-processing (with operators of cdo, Climate Data Operators, software) Global State Vector (GSV) extracted modeled data (over latitude-longitude domain) for selected meteorological variables into hourly time-slots/slices (00...23 UTCs);
* Extracting (using sql with odp\_api software) selected meteorological variables at hourly time-slices for observations (from ODB) from: (i) ground-based synoptic stations at fixed geographical latitude-longitude locations on surface; (ii) radiosounding stations and satellites at fixed latitude-longitude locations on surface and on multiple pressure levels;
* Extracting and applying appropriate observation operator (using cdo operators - for synop; & expecting/using polytope - for radiosounding and satellite) to modeled data for the same time-slices for same selected meteorological variables into corresponding locations/points of (synop, radiosounding, satellite) observations, with end result - augmented ODB;
* Calculating/ producing relevant statistics for monitoring such as quantile rank histogram statistics and plots.

## **Contributing**

OBSALL Apps is developed under the European Union Contract DE\_340\_CSC - Destination Earth Programme Climate Adaptation Digital Twin (Climate DT). Contributions to the project are welcome and can be made through the OBSALL repository at GitLab hosted by BSC.

Please, see and follow guidelines in Chapter “CONTRIBUTING TO OBSALL”.

# **CHAPTER 2: INSTALLATION**

## **Prerequisites**

Before installing OBSALL Apps, ensure that you have the following:

• Access to Lumi HPC for the DestinE project; so, you need to be registered with user account.

• Git: OBSALL is hosted on GitLab, and you will need Git to clone the repository.

• Access to GitLab hosted by BSC (Barcelona Supercomputing Center); so, you need to be registered with user account. Contact Albert Vilamiro ([albert.vilamiro@bsc.es](mailto:albert.vilamiro@bsc.es)) to authorize registration and project you would like to join. Once your request is validated, an email will be sent with further instructions to follow, and login at https://earth.bsc.es/gitlab/users/sign\_in.

• Access to repository for OBSALL Apps; so, you need to contact Albert Vila ([albert.vila@bsc.es](mailto:albert.vila@bsc.es)) with CC to Pablo Ortega (pablo.ortega@bsc.es) requesting access to the "obsall” repository of the de340 project.

## **Installation on Lumi HPC**

### ***2.2.1. Cloning/ Copying***

* **Clone/ copy** the OBSALL Apps ("obsall" repository) to your local directory via Git using:

**cd your\_local\_directory**

**git clone https://earth.bsc.es/gitlab/digital-twins/de\_340/obsall.git**

**cd obsall**

**git checkout newbranch**

This will clone the "newbranch"-branch of the "obsall" repository.

OR you may copy tar-file containing “obsall” repository

**cd your\_local\_directory**

**cp /projappl/project\_465000454/ama/ARC\_OBSALL/arc\_obsall\_apps.tar .**

**tar -xvf arc\_obsall\_apps.tar**

**cd obsall**

This will copy the "newbranch"-branch of the "obsall" repository.

The basic content of the “obsall” directory is:

|  |  |
| --- | --- |
| drwxrws--- FIGSEXAMPLES  drwxrws--- .git  drwxrws--- RADSOUND  -rw-rw---- README.md  -rw-rw---- request\_radsound.yml  -rw-rw---- request\_synop.yml  -rw-rw---- run\_obsall.py  -rwxrw---- run\_obsall.sh  drwxrws--- SATELLITE  drwxrws--- SYNOP | - illustration material (figures, demo video, slides) for OBSALL Apps  - contains all the log messages, author information, and other information required to rebuild any version or branch of the project  - all scripts/programs/etc. for the TEMP/ RADSOUND Part  - basic overall description of the Apps  - request for TEMP Part (gsv extract modelled data for radionsonding obs)  - request for SYNOP Part (gsv extract modelled data for synop obs)  - python script to execute Apps  - bash script to execute Apps  - all scripts/programs/etc. for the AMSU-A/ SATELLITE Part  - all scripts/programs/etc. for the SYNOP Part |

### ***2.2.2. Setting environment***

* **Set environment** (**load necessary modules** required for SYNOP, RADSOUND, and SATELLITE Parts) in order to run OBSALL application on LUMI: execute bash-script which is containing list of required modules (see below): **./load\_modules.sh**

**module use /project/project\_465000454/devaraju/modules/LUMI/23.03/C**

**module load LUMI/23.03**

**module load partition/C**

**module load PrgEnv-gnu**

**module load ecCodes/2.32.0-cpeCray-23.03.lua**

**module load odb\_api/0.18.1-cpeCray-23.03.lua**

**module load python-climatedt/3.11.3-cpeCray-23.03.lua**

**module load pyfdb/0.0.2-cpeCray-23.03.lua**

**module load cray-hdf5/1.12.2.3**

**module load cray-netcdf/4.9.0.3**

**module load rttov/13.2**

**module load radsim/3.2**

**module load fdb/5.11.94-cpeCray-23.03.lua**

**module load eckit/1.25.0-cpeCray-23.03.lua**

**module load metkit/1.11.0-cpeCray-23.03.lua**

This will load all above-listed modules.

### ***2.2.3. Executing application***

* **Execute** the OBSALL Apps by running script: **python run\_obsall.py**

This python script will subsequently run the 3 Parts of OBSALL Apps

#!/scratch/project\_465000454/devaraju/SW/LUMI-23.03/C/python-climatedt/bin/python

# OBSALL Apps (3 parts: SYNOP, TEMP, AMSU-A observations)

# Import required libraries

import sys

import subprocess

## --- Processing ground-based observations (SYNOP)

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print('DestinE Climate Digital Twin - OBSALL Apps')

print('--- Processing ground-based observations (SYNOP)')

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

command\_synop\_run = "cd SYNOP; pwd; ./main\_synop.sh; exit 0"

subprocess.run(command\_synop\_run, shell=True, check=True, executable="/bin/bash")

## --- Processing radiosounding observations (TEMP)

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print('DestinE Climate Digital Twin - OBSALL Apps')

print('--- Processing radiosounding-based observations (TEMP)')

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

command\_radsound\_run = "cd RADSOUND; pwd; ./main\_radsound.sh; exit 0"

subprocess.run(command\_radsound\_run, shell=True, check=True, executable="/bin/bash")

## Processing satellite observations (AMSU-A)

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print('DestinE Climate Digital Twin - OBSALL Apps')

print('--- Processing satelitte-based observations (AMSU-A)')

print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

command\_satellite\_run = "cd SATELLITE; pwd; ./main\_amsua.sh; exit 0"

subprocess.run(command\_satellite\_run, shell=True, check=True, executable="/bin/bash")

sys.exit(0)

### ***2.2.4. Requesting modelled data***

* **For SYNOP Part the Data request (sfc)**: gsv extracting modelled data at sfc-level: 2t & other available meteorological variables.
* **For TEMP Part the Data request (sfc+pl)**: gsv extracting modelled data at (1) sfc-level: 2t, and (2) pl-level: t (at 850 hPa).
* **For AMSU-A Part the Data request (sfc+pl+o2d)**: gsv extracting modelled data at (1) sfc-level: 2t, 2d, sp, skt,10u, 10v & z, lsm (n/a – IFS; tmp. in static grib file); (2) pl-levels: t, q, clwc, z (at 1000, 925, 850, 700, 600, 500, 400, 300, 250, 200, 150, 100, 70, 50, 30, 20, 10, 5, 1 hPa) – 19 pl-levels; and (3) o2d-level: avg\_siconc – from NEMO model (or ci – from IFS, n/a now).
* **Data request combined (sfc, pl, and o2d -levels)**: yml-file is needed for extracting selected modeled data from IFS and NEMO models at sfc-, pl-, and o2d-levels. The example (see at at <https://earth.bsc.es/gitlab/digital-twins/de_340/obsall/-/tree/newbranch>) is given as part of the **mother\_request.yml**-file for extraction of all needed variables for synop (only at sfc-level), radiosounding (at both sfc- and pl-levels) and satellite (at sfc, pl-, and o2d-levels) parts of OBSALL Apps.

See details info on request of modeled data for 3 Parts of Apps:

**GSVREQUEST**: dataset, class, type, expver, stream, activity, resolution, generation, realization, experiment, model, levtype, date, time, param, grid, method

**OPAREQUEST**: variable, stat, stat\_freq, output\_freq, time\_step, save, checkpoint, checkpoint\_filepath, save\_filepath

* **Container** for OBSALL Apps (to be followed soon)

# **CHAPTER 3: MAIN COMPONENTS/ PARTS OF OBSALL**

## **Part SYNOP (ground-based/surface observations)**

**main\_synop.sh** - main bash-script to run Apps for observational data for synop stations.

**INPUT: LIST OF NECESSARY PARAMETERS**:

1. **varMETnum** – meteorological variable code in ODB (for example, 39 is for 2 metre air temperature in ODB)
2. **varMETstr** – corresponding meteorological variable code in FDB (for example, 2t is in FDB)
3. **lon\_val\_min** – western boundary of selected geographical domain (for example, 20)
4. **lon\_val\_max** – eastern boundary of selected geographical domain (for example, 25)
5. **lat\_val\_min** – southern boundary of selected geographical domain (for example, 60)
6. **lat\_val\_max** – northern boundary of selected geographical domain (for example, 65)

This script calls subsequently to

* **gsv\_mod\_data.sh** – to pre-process necessary extracted modeled data with gsv\_interface
* **synop\_obs.sh** – to extract observation data (from ODB) at geographical locations (latitude, longitude) for each synop station
* **synop\_mod.sh** – to extract and interpolate modeled data at exact locations of synop stations, and to add interpolated modelled data to ODB
* **graph\_mod\_obs.py** – to plot differences between observation and modelled data (“for self-control”; can be ignored)
* **synop\_stats.sh** – to calculate relevant statistics and produce a series of plot (see **Figures 1abc**)

**PROCESSING STEPS**:

* **gsv\_mod\_data.sh** - bash-script to run pre-processing (using **cdo** operators) of modelled data extracted with gsv interface over global domain for selected meteorological variables (measured at synop station) into hourly time-slices and saving (temporary) to separate hourly nc-files.
* **synop\_obs.sh** - bash-script to run reading and extraction (using **sql** with **odb\_api** software) of selected meteorological variables at hourly time-slices for observations from ground-based synop stations at fixed at the surface geographical (latitude, longitude) locations over selected geographical domain (or over the globe) and saving (temporary) to dat-file. Note, that list of available meteorological variables in open multi-year data ODB includes: 91 - total amount of clouds; 108 - sea level pressure; 80 - 1-hour precipitation; 58 - relative humidity; 999 - 10-minute precipitation intensity; 71 - snow depth; 39 - 2m temperature; 40 - dew point temperature; 62 - visibility; 111 - wind direction; 261 - maximum wind gust in last 10 minutes; 221 -surface wind speed.
* **synop\_mod.sh** - bash-script to run reading, extraction and interpolation (using **cdo** operators) of modelled data for selected meteorological variables at the same hourly time-slices to geographical (latitude, longitude) locations of synop stations, and saving (temporary) to dat-file, and adding (using **import** with **odb\_api**) such data to ODB. Interpolation is preceded by constructing unstructured grid based on locations of synop stations and calculating weights for such grid.
* **graph\_mod\_obs.py** - python-script to run an internal self-control in calculating differences between observed and modelled data for locations of synop stations.
* **synop\_stats.sh** – to calculate relevant statistics and produce a series of plots (see section 3.2 for details).

**PATH TO ALL FILES ABOVE IS: obsall/SYNOP**

## **Monitoring for SYNOP Part**

**synop\_stats.sh** - bash-script to run (additional bash, fortran, python scripts) and calculate necessary relevant statistics and produce a series of relevant plots for synop stations.

This script calls subsequently to

* **produce\_rank\_histograms\_all\_stations.sh** – to access necessary data, calculate quantile rank histograms and their summary statistics for each synop station for 00, 06, 12, and 18 UTCs (see section 3.2.1).
* **produce\_standard\_plots\_all\_stations.sh** – to access required data and to call/plot quantiles rank histogram for each synop station (see section 3.2.2).
* **summary\_rank\_histograms\_all\_stations.sh** – to accessing required data, to calculate rank histogram summary statistics over all synop stations, to call/plot p-values over geographical domain for all stations and rank histogram summary statistics based on all synop stations (see section 3.2.3).

### ***3.2.1. Rank histograms for all synop stations for 00, 06, 12, and 18 UTCs***

(with **produce\_rank\_histograms\_all\_stations.sh**, using **rank\_histograms\_one\_station.f95** and **sql/import** with **odb\_api**)

**FILE (BASH-SCRIPT): obsall/SYNOP/STATS/produce\_rank\_histograms\_all\_stations.sh** is used for calculation of quantile space rank histograms and their summary statistics (Mean Square Deviation, MSD and p-value) for all synop stations for 00, 06, 12, and 18 UTCs.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **synop\_stats.sh** script
2. Accessing raw data from model simulations, quantiles and bootstrap MSD statistics
3. Accessing file with list of synop stations and their geographical coordinates
4. Compiling fortran program **rank\_histograms\_one\_station.f95** to calculate rank histograms
5. Initiating loop over all synop stations
6. Selecting the simulation data for synop station
7. Selecting the quantiles for synop station
8. Selecting the MSD for synop station
9. Producing the rank histogram for synop station for 00, 06, 12, and 18 UTCs
10. Converting the rank histogram file to ODB format
11. Combining the rank histogram data to all synop stations in a single ODB file

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable** – meteorological variable code in ODB (for example, 39 is for 2 metre air temperature in ODB)
2. **year1** – first year (for example, 2020)
3. **year2** – last year (for example, 2020)
4. **month1** – first month (for example, 01)
5. **month2** – last month (for example, 12)

*Note:* ***${period}*** *is calculated based on* ***year1, year2, month1, month2***

**LIST OF INPUT DATA REQUIRED:**

1. file incl. list of synop stations (it is assumed that the list of such stations has been pre-produced)

**obsall/SYNOP/STATS/list\_of\_stations\_${variable}.txt**

1. path to location of raw simulation data at coordinates of synop stations

**sim\_dir=/scratch/project\_465000454/ama/open\_data\_ODB**

1. path to location of file incl. quantiles as a function of time of year

**quant\_dir=/scratch/project\_465000454/ama/STATDATASYNOP/quantiles**

1. path to location of file incl. rank histogram bootstrap MSD values

**bootstrap\_dir=/scratch/project\_465000454/ama/STATDATASYNOP/bootstrap\_statistics/${variable}\_${period}**

**OUTPUT FOR RESULTS:**

1. path to store calculated output for each synop station in separate files

**outdir=** **obsall/SYNOP/STATS/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**outfile=${outdir}/rank\_histograms\_${variable}\_${station}\_${year1}${month1}-${year2}${month2}**

*Note:* ***${station}*** *is read from file* ***list\_of\_stations\_${variable}.txt***

**FILE (FORTRAN PROGRAM): obsall/SYNOP/STATS/fortran-programs/rank\_histograms\_one\_station.f95** is called and compiled from **produce\_rank\_histograms\_all\_stations.sh** and calculates for each synop station the frequencies of quantiles having output incl. station code/ID, longitude, latitude, UTC hour, number of realizations for this hour, MSD for the quantile frequencies, p-value of MSD (=fraction of bootstrap realizations with larger or equal MSD), and quantile frequencies (on scale 0-1) from 0-1% ... 99-100%.

### ***3.2.2. Standard plots for each synop station***

(with **produce\_standard\_plots\_all\_stations.sh** using **plot\_quantiles\_rankhist\_synop.py** and **sql** with **odb\_api**; see an example of such plot in **Figure 1a**)

**FILE (BASH-SCRIPT):** **obsall/SYNOP/STATS/produce\_standard\_plots\_all\_stations.sh** is used for accessing required data and calling for plotting quantiles rank histogram for each individual synop station.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **synop\_stats.sh** script
2. Accessing raw data from model simulations, observations, quantiles, and calculated rank histogram data
3. Accessing file with list of synop stations and their geographical coordinates
4. Initiating loop over all synop stations
5. Selecting the simulation data for synop station
6. Selecting the rank histogram data for synop station
7. Selecting the quantiles for synop station
8. Calling python script **plot\_quantiles\_rankhist\_synop.py** to plot quantiles rank histogram for synop station

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable** – meteorological variable code in ODB (for example, 39 is for 2 metre air temperature in ODB)
2. **year1** – first year (for example, 2020)
3. **year2** – last year (for example, 2020)
4. **month1** – first month (for example, 01)
5. **month2** – last month (for example, 12)

**LIST OF INPUT DATA REQUIRED:**

1. file incl. list of synop stations (it is assumed that the list of such stations has been pre-produced)

**obsall/SYNOP/STATS/list\_of\_stations\_${variable}.txt**

1. path to location of raw simulation data at coordinates of synop stations

**sim\_dir=/scratch/project\_465000454/ama/open\_data\_ODB**

1. path to location of file incl. quantiles as a function of time of year

**quant\_dir=/scratch/project\_465000454/ama/STATDATASYNOP/quantiles**

1. path to location of file incl. computed rank histogram data

**rh\_dir=** **obsall/SYNOP/STATS/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**OUTPUT FOR RESULTS (PLOTS):**

1. path to store calculated output for each synop station in separate files

**figure\_dir=obsall/SYNOP/STATS/figures/standard\_plots\_${variable}\_${year1}${month1}-${year2}${month2}**

**fig\_name=${figure\_dir}/standard\_plot\_${variable}\_${station}\_${year1}${month1}-${year2}${month2}\_python.png**

*Note:* ***${station}*** *is read from file* ***list\_of\_stations\_${variable}.txt***

**FILE (PYTHON PROGRAM): obsall/SYNOP/STATS/python/plot\_quantiles\_rankhist\_synop.py** produces individual plot (see, for example, **Figure 1a** “Quantiles/ time series data (00, 06, 12, and 18 UTCs) and rank histogram (combination of 00, 06, 12, and 18 UTCs) synop stationID”) for each synop station. Python program uses passed list of parameters: **quantiles\_data, rank\_data, time\_series\_data, variable, station\_id, lon, lat, year\_beg, year\_end, month\_beg, month\_end, figname** from the **produce\_standard\_plots\_all\_stations.sh** bash-script.



***Figure 1a****: Plot for 2 metre air temperature (T2m) for period of 1 Jan - 31 Dec 2020: Quantiles/ time series data (00, 06, 12, and 18 UTCs) and rank histogram (combination of 00, 06, 12, and 18 UTCs) for synop station 126736.*

### ***3.2.3. Summary rank histograms for all synop stations***

(with **summary\_rank\_histograms\_all\_stations.sh** using **rank\_histogram\_summary\_statistics.f95**, **plot\_p\_values\_map\_synop.py**, **plot\_rankhist\_sum\_all\_stations\_synop.py** and **sql** with **odb\_api**; see examples of such plots in **Figure 1b** and **Figure 1c**).

**FILE (BASH-SCRIPT): obsall/SYNOP/STATS/summary\_rank\_histograms\_all\_stations.sh** is used for accessing required data, calculating rank histogram summary statistics over all synop stations, calling for plotting p-values over geographical domain for all stations, and calling for plotting rank histogram summary statistics based on all synop stations.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **synop\_stats.sh** script
2. Accessing rank histogram data
3. Compiling fortran program **rank\_histogram\_summary\_statistics.f95** to calculate rank histogram summary statistics
4. Calling python script **plot\_p\_values\_map\_synop.py** to plot p-values for all synop stations over geographical domain
5. Calling python script **plot\_rankhist\_sum\_all\_stations\_synop.py** to plot rank histogram summary statistics based on all synop stations

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable** – meteorological variable code in ODB (for example, 39 is for 2 metre air temperature in ODB)
2. **year1** – first year (for example, 2020)
3. **year2** – last year (for example, 2020)
4. **month1** – first month (for example, 01)
5. **month2** – last month (for example, 12)

**LIST OF INPUT DATA REQUIRED:**

1. path to location of file incl. computed rank histogram data

**rh\_dir=** **obsall/SYNOP/STATS/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**OUTPUT FOR RESULTS (PLOTS):**

1. path to store plotted output (based on all synop stations) in separate files

**figure\_dir=** **obsall/SYNOP/figures**

**figure\_dir/p-value\_map\_${variable}\_${year1}${month1}-${year2}${month2}.png**

**figure\_dir/rank-hist-sumstats\_${variable}\_${year1}${month1}-${year2}${month2}.png**

**FILE (PYTHON PROGRAM): obsall/SYNOP/STATS/python/plot\_p\_values\_map\_synop.py** produces a single plot (see, for example, **Figure 1b** “P-values for 2 metre air temperature quantiles”). Python program uses passed list of parameters: **p\_values\_file, p\_values** from the **summary\_rank\_histograms\_all\_stations.sh** bash-script.

**FILE (PYTHON PROGRAM): obsall/SYNOP/STATS/pyhon/plot\_rankhist\_sum\_all\_stations\_synop.py** produces a single plot (see, for example, **Figure 1c** “Rank histogram summary statistics for 2 metre air temperature: normalized p-value frequencies and normalized quantile frequencies”). Python program uses passed list of parameters: **p\_freq, q\_freq, number\_of\_stations, p01, p1, p5, max\_freq\_p, and max\_freq\_q** from the **summary\_rank\_histograms\_all\_stations.sh** bash-script.

A map of the world with different colored dots

Description automatically generated

***Figure 1b****: Plot for 2 metre air temperature (2t) for period of 1 Jan - 31 Dec 2020: P-values for 2t quantiles.*

A close-up of several graphs

Description automatically generated

***Figure 1c****: Plot for 2 metre air temperature (2t) for period of 1 Jan - 31 Dec 2020: Rank histogram summary statistics for 2t: normalized p-value frequencies and normalized quantile frequencies.*

## **Part TEMP (radiosounding observations)**

**main\_radsound.sh** - main bash-script to run Apps for observational data for radiosounding stations.

**INPUT: LIST OF NECESSARY PARAMETERS**:

1. **varMETnum** – meteorological variable code in ODB (for example, 39 is for 2 metre air temperature in ODB)
2. **varMETstr** – corresponding meteorological variable code in FDB (for example, 2t is in FDB)
3. **lon\_val\_min** – western boundary of selected geographical domain (for example, 20)
4. **lon\_val\_max** – eastern boundary of selected geographical domain (for example, 25)
5. **lat\_val\_min** – southern boundary of selected geographical domain (for example, 60)
6. **lat\_val\_max** – northern boundary of selected geographical domain (for example, 65)

This script calls subsequently to

* **gsv\_radsound\_mod\_data.sh** – to pre-process necessary extracted modeled data with gsv\_interface
* **radsound\_obs.sh** – to extract observation data (from ODB) at geographical locations (latitude, longitude, pressure level) for each radiosounding station
* **radsound\_mod.sh** – to extract and interpolate modeled data at exact locations of radiosounding stations, and to add interpolated modelled data to ODB
* **radsound\_stats.sh** – to calculate relevant statistics and produce a series of plot (see **Figures 2abc**)

**PROCESSING STEPS**:

* **gsv\_radsound\_mod\_data.sh** - bash-script to run pre-processing (using **cdo** operators) of modelled data extracted with gsv interface over global domain for air temperature at 2 metre (T2) and at 850 hPa pressure level (T850) into time-slices at 00 and 12 UTCs, calculating difference (T2-T850) between these two temperatures, and saving (temporary) to separate nc-files at 00 and 12 UTC time-slices.
* **radsound\_obs.sh** - bash-script to run reading and extraction (using **sql** with **odb\_api** software) of above-mentioned air temperatures at 12 hourly time-slices for radiosounding observations at surface and at altitude at geographical (latitude, longitude, pressure level) locations of radiosounding stations over selected geographical domain (or over the globe), calculating difference between two temperatures, and saving (temporary) to dat-file.
* **radsound\_mod.sh** - bash-script to run reading, extraction and interpolation (**cdo**/ later using **polytope**) of modelled data for air temperature at surface and at altitude at the same time-slices to geographical (latitude, longitude, pressure level) locations of radiosounding stations, calculating difference between two temperatures, and saving (temporary) to dat-file, and adding (using import with **odb\_api**) such data to ODB.
* **radsound\_stats.sh** – to calculate relevant statistics and produce a series of plots (see section 3.4 for details).

**PATH TO ALL FILES ABOVE IS: obsall/RADSOUND**

## **Monitoring for TEMP Part**

**radsound\_stats.sh** - bash-script to run (additional bash, fortran, python scripts) and calculate necessary relevant statistics and produce a series of relevant plots for radiosounding stations.

This script calls subsequently to

* **produce\_rank\_histograms\_all\_stations.sh** – to access necessary data, calculate quantile rank histograms and their summary statistics for each radiosounding station for 00 and 12 UTCs (see section 3.4.1).
* **produce\_standard\_plots\_all\_stations.sh** – to access required data and to call/plot quantiles rank histogram for each radiosounding station (see section 3.4.2).
* **summary\_rank\_histograms\_all\_stations.sh** – to accessing required data, to calculate rank histogram summary statistics over all radiosounding stations, to call/plot p-values over geographical domain for all stations and rank histogram summary statistics based on all radiosounding stations (see section 3.4.3).

### ***3.4.1. Rank histograms for all radiosounding stations for 00 and 12 UTCs***

(with **produce\_rank\_histograms\_all\_stations.sh**, using **rank\_histograms\_one\_station.f95** and **sql/import** with **odb\_api**)

**FILE (BASH-SCRIPT): obsall/RADSOUND/STATRS/produce\_rank\_histograms\_all\_stations.sh** is used for calculation of quantile space rank histograms and their summary statistics (Mean Square Deviation, MSD and p-value) for all radiosounding stations for 00 and 12 UTCs.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **synop\_stats.sh** script
2. Accessing raw data from model simulations, quantiles and bootstrap MSD statistics
3. Accessing file with list of radiosounding stations and their geographical coordinates
4. Compiling fortran program **rank\_histograms\_one\_station.f95** to calculate rank histograms
5. Initiating loop over all radiosounding stations
6. Selecting the simulation data for radiosounding station
7. Selecting the quantiles for radiosounding station
8. Selecting the MSD for radiosounding station
9. Producing the rank histogram for radiosounding station for 00 and 12 UTCs
10. Converting the rank histogram file to ODB format
11. Combining the rank histogram data to all radiosounding stations in a single ODB file

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable** – “**T2-T850**” (difference between the air temperatures at 2 metre and 850 hPa level)

*Note:* ***T2-T850*** *is only one variable used for radiosounding data*

1. **year1** – first year (for example, 2010)
2. **year2** – last year (for example, 2010)
3. **month1** – first month (for example, 01)
4. **month2** – last month (for example, 12)

*Note:* ***${period}*** *is calculated based on* ***year1, year2, month1, month2***

**LIST OF INPUT DATA REQUIRED:**

1. file incl. list of radiosounding stations (it is assumed that the list of such stations has been pre-produced & includes stations with, at least, 1000 observation data points)

**obsall/RADSOUND/STATRS/list\_of\_valid\_radsound\_stations.txt**

1. path to location of raw simulation data at coordinates of radiosounding stations

**sim\_dir= scratch/project\_465000454/ama/STATDATARADSOUND/RHARM\_vertical\_differences\_odb**

1. path to location of file incl. quantiles as a function of time of year

**quant\_dir=/scratch/project\_465000454/ama/STATDATARADSOUND/quantiles**

1. path to location of file incl. rank histogram bootstrap MSD values

**bootstrap\_dir=/scratch/project\_465000454/ama/STATDATARADSOUND/bootstrap\_statistics/${variable}\_${period}**

**OUTPUT FOR RESULTS:**

1. path to store calculated output for each radiosounding station in separate files

**outdir=** **obsall/RADSOUND/STATRS/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**outfile=${outdir}/rank\_histograms\_${variable}\_${station}\_${year1}${month1}-${year2}${month2}**

*Note:* ***${station}*** *is read from file* ***list\_of\_valid\_radsound\_stations.txt***

**FILE (FORTRAN PROGRAM): obsall/RADSOUND/STATRS/fortran-programs/rank\_histograms\_one\_ station.f95** is called and compiled from **produce\_rank\_histograms\_all\_stations.sh** and calculates for each synop station the frequencies of quantiles having output incl. station code/ID, longitude, latitude, UTC hour, number of realizations for this hour, MSD for the quantile frequencies, p-value of MSD (=fraction of bootstrap realizations with larger or equal MSD), and quantile frequencies (on scale 0-1) from 0-1% ... 99-100%.

### ***3.4.2. Standard plots for each radiosounding station***

(with **produce\_standard\_plots\_all\_stations.sh** using **plot\_quantiles\_rankhist\_temp.py** and **sql** with **odb\_api**; see an example of such plot in **Figure 2a**)

**FILE (BASH-SCRIPT): obsall/RADSOUND/STATRS/produce\_standard\_plots\_all\_stations.sh** is used for accessing required data and calling for plotting quantiles rank histogram for each individual radiosounding station.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **radsound\_stats.sh** script
2. Accessing raw data from model simulations, observations, quantiles, and calculated rank histogram data
3. Accessing file with list of radiosounding stations and their geographical coordinates
4. Initiating loop over all radiosounding stations
5. Selecting the simulation data for radiosounding station
6. Selecting the rank histogram data for radiosounding station
7. Selecting the quantiles for radiosounding station
8. Calling python script **plot\_quantiles\_rankhist\_temp.py** to plot quantiles rank histogram for radiosounding station

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable – “T2-T850”** (difference between the air temperatures at 2 metre and 850 hPa level)

*Note: T2-T850 is only one variable used for radiosounding data*

1. **year1** – first year (for example, 2010)
2. **year2** – last year (for example, 2010)
3. **month1** – first month (for example, 01)
4. **month2** – last month (for example, 12)

**LIST OF INPUT DATA REQUIRED:**

1. file incl. list of radiosounding stations (it is assumed that the list of such stations has been pre-produced & includes stations with, at least, 1000 observation data points)

**obsall/RADSOUND/STATRS/list\_of\_valid\_radsound\_stations.txt**

1. path to location of raw simulation data at coordinates of radiosounding stations

**sim\_dir= /scratch/project\_465000454/ama/STATDATARADSOUND/RHARM\_vertical\_differences\_odb**

1. path to location of file incl. quantiles as a function of time of year

**quant\_dir=/scratch/project\_465000454/ama/STATDATARADSOUND**

1. path to location of file incl. computed rank histogram data

**rh\_dir=** **obsall/RADSOUND/STATRS/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**OUTPUT FOR RESULTS (PLOTS):**

1. path to store calculated output for each radiosounding station in separate files

**figure\_dir=obsall/RADSOUND/STATRS/figures/standard\_plots\_${variable}\_${year1}${month1}-${year2}${month2}**

**fig\_name=${figure\_dir}/standard\_plot\_${variable}\_${station}\_${year1}${month1}-${year2}${month2}.png**

*Note:* ***${station}*** *is read from file* ***list\_of\_valid\_radsound\_stations.txt***

**FILE (PYTHON PROGRAM): obsall/RADSOUND/STATRS/python/plot\_quantiles\_rankhist\_temp.py** produces individual plot (see, for example, **Figure 2a** “Quantiles/ time series data and rank histogram (00 and 12 UTCs) radiosounding stationID”) for each radiosounding station. Python program uses passed list of parameters: **quantiles\_data, rank\_data, time\_series\_data, station\_id, lon, lat, year\_beg, year\_end, month\_beg, month\_end, fig\_name** from the **produce\_standard\_plots\_all\_stations.sh** bash-script.

A close-up of a graph

Description automatically generated

***Figure 2a****: Plots for difference in air temperatures at 2 metre and at 850 hPa pressure level (T2-T850) for period of 1 Jan – 31 Dec 2010: Quantiles/ time series data and rank histogram (00 and 12 UTCs) for radiosounding station 51431.*

### ***3.4.3. Summary rank histograms for all radiosounding stations***

(with **summary\_rank\_histograms\_all\_stations.sh** using **rank\_histogram\_summary\_statistics.f95**, **plot\_p\_values\_map\_temp.py**, **plot\_rankhist\_sum\_all\_stations\_temp.py** and **sql** with **odb\_api**; see examples of such plots in **Figure 1b** and **Figure 1c**).

**FILE (BASH-SCRIPT): obsall/RADSOUND/STATRS/summary\_rank\_histograms\_all\_stations.sh** is used for accessing required data, calculating rank histogram summary statistics over all radiosounding stations, calling for plotting p-values over geographical domain for all stations, and calling for plotting rank histogram summary statistics based on all radiosounding stations.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **radsound\_stats.sh** script
2. Accessing rank histogram data
3. Compiling fortran program **rank\_histogram\_summary\_statistics.f95** to calculate rank histogram summary statistics for 00 and 12 UTCs
4. Calling python script **plot\_p\_values\_map\_temp.py** to plot p-values for all radiosounding stations over geographical domain for 00 and 12 UCTs
5. Calling python script **plot\_rankhist\_sum\_all\_stations\_temp.py** to plot rank histogram summary statistics based on all radiosounding stations or 00 and 12 UTCS

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable** – “T2-T850” (difference between the air temperatures at 2 metre and 850 hPa level)

*Note: T2-T850 is only one variable used for radiosounding data*

1. **year1** – first year (for example, 2010)
2. **year2** – last year (for example, 2010)
3. **month1** – first month (for example, 01)
4. **month2** – last month (for example, 12)

**LIST OF INPUT DATA REQUIRED:**

1. path to location of file incl. computed rank histogram data

**rh\_dir=** **obsall/RADSOUND/STATRS/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**OUTPUT FOR RESULTS (PLOTS):**

1. path to store plotted output (based on all radiosounding stations) in separate files

**figure\_dir=** **obsall/RADSOUND/STATRS/figures**

**figure\_dir/p-values\_rank-histograms\_${variable}\_${year1}${month1}-${year2}${month2}.png**

**figure\_dir/summary\_plot\_rank-histograms\_${variable}\_${year1}${month1}-${year2}${month2}.png**

**FILE (PYTHON PROGRAM): obsall/RADSOUND/STATRS/python/plot\_p\_values\_map\_temp.py** produces a single plot (see, for example, ***Figure 2b*** “P-values for T2-T850 quantiles”). Python program uses passed list of parameters: **p\_values\_file, p\_values\_00, p\_values\_12** from the **summary\_rank\_histograms\_all\_stations.sh** bash-script.

**FILE (PYTHON PROGRAM): obsall/RADSOUND/STATRS/python/plot\_rankhist\_sum\_all\_stations\_temp.py** produces a single plot (see, for example, ***Figure 2c***“Rank histogram summary statistics for T2-T850 for 00 and 12 UTCs: normalized p-value frequencies and normalized quantile frequencies”). Python program uses passed list of parameters: **p\_freq\_00, p\_freq\_12, q\_freq\_00, q\_freq\_12, number\_of\_stations, p01\_00, p1\_00, p5\_00, p01\_12, p1\_12, p5\_12, p\_ylim, q\_ylim, figname** from the **summary\_rank\_histograms\_all\_stations.sh** bash-script.

A map of the world with different colored dots

Description automatically generated

***Figure 2b****: Plots for difference in air temperatures at 2 metre and at 850 hPa pressure level (T2-T850) for period of 1 Jan - 31 Dec 2010: P-values for T2-T850 quantiles.*

A close-up of several graphs

Description automatically generated

***Figure 2c***: *Plots for difference in air temperatures at 2 metre and at 850 hPa pressure level (T2-T850) for period of 1 Jan - 31 Dec 2010: Rank histogram summary statistics for 00 and 12 UTCs: normalized p-value frequencies and normalized quantile frequencies.*

## **Part AMSUA-A (satellite observations)**

**main\_amsua.sh** - main bash-script to run Apps for observational data for satellites.

**INPUT: LIST OF NECESSARY PARAMETERS**:

1. **varMETname** – meteorological variable name (btmp – brightness temperature)
2. **varMETnum** – meteorological variable code (194 for btmp)
3. **channel** – Satellite AMSU-A channel number (5)

This script calls subsequently to

* **set\_env.sh** – bash-script which defines variables and functions, and required modules to be loaded and used in the workflow.
* **gsv\_mod\_data.sh** – to pre-process necessary extracted modeled data from sfc-, pl-, and o2d-levels with gsv\_interface over area, interpolate from pl- to model-levels
* **amsua\_obs.sh** – to prepare metadata for computation of AMSU-A model counterparts for each area
* **amsua\_mod.sh** – to run the Radiance Simulator and to produce a model counterpart, and add interpolated modelled data to ODB
* **amsua\_stats.sh** – to calculate relevant statistics and produce a series of plot (see **Figures 3abc**)

**PROCESSING STEPS**:

* **gsv\_mod\_data.sh** – bash-script to run pre-processing (using **cdo** operators; in later versions - **polytope**) of 2D on surface (sfc) and 3D modelled data on pressure levels (pl) extracted with gsv interface over global domain. The **Radiance Simulator** (**using convert\_pl2ml.py**) interpolates modelled data from pl-levels to “model”-levels (note, workflow will initially be demonstrated on a single selected location i.e., around a point of 70.0N, 45.0E in the Barents Sea; and later geographically expanded using polytope) and preprocessed data saved in (temporary) hourly GRIB-files. Required 3D variables are: t - temperature (130), q - specific humidity (133), clwc - specific liquid water content (246); and 2D variables are: skt - skin temperature (235), ci - sea ice fraction (31), lsm - land-sea mask (172), sp - surface pressure (134), 10u and 10v - U and V components of wind speed at 10 m (165 and 166), 2t - air temperature at 2 m (167), 2d - dewpoint temperature at 2 m (168), and z - geopotential (for orography) (129).
* **amsua\_obs.sh** - bash-script to prepare a metadata file for computation of AMSU-A model counterparts from preprocessed hourly GSV data. It runs depending on the type of climate simulation: (i) reality following data assimilation-like, and (ii) free running climate simulation. For (i) the one-to-one model counterparts can be produced, so actual metadata from AMSU-A observations in the vicinity of the selected location are extracted from ODB. This is done with **filter\_data.py**. For (ii) the model output can be compared to observations only in climatological terms, so metadata is written only for the centre point of the selected area.
* **amsua\_mod.sh** - bash-script to run the **Radiance Simulator** tool with pre-described control configuration parameters and to produce a model counterpart (file in netcdf-format), which (using **convert\_netcdf2txt.py**) is converted to (temporary) dat-file and added (using **import** with **odb\_api**) to ODB.
* **amsua\_stats.sh** – to calculate relevant statistics and produce a series of plots (see section 3.6 for details).

**PATH TO ALL FILES ABOVE IS: obsall/SATELLITE/STATST**

## **Monitoring for AMSU-A Part**

**amsua\_stats.sh** - bash-script to run (additional bash, fortran, python scripts) and calculate necessary relevant statistics and produce a series of relevant plots for selected areas, for example, are “a00” (with a size of about 100x100 km and assigned to exact geographical locations, i.e., to the center point of the area)

This script calls subsequently to

* **produce\_rank\_histograms\_all\_locations.sh** – to access necessary data, calculate quantile rank histograms and their summary statistics for all areas from satellite observations for daily mean values of brightness temperature (see section 3.6.1).
* **produce\_standard\_plots\_all\_locations.sh** – to access required data and to call/plot quantiles rank histogram for each area (see section 3.6.2).
* **summary\_rank\_histograms\_all\_locations.sh** – to accessing required data, to calculate rank histogram summary statistics over all areas, to call/plot p-values over geographical domain for all areas and rank histogram summary statistics based on all areas (see section 3.6.3).

### ***3.6.1. Rank histograms for all areas for daily mean***

(with **produce\_rank\_histograms\_all\_locations.sh**, using **rank\_histograms\_one\_location.f95** and **daymean.py** and **sql/ import** with **odb\_api**),

**FILE (BASH-SCRIPT): obsall/SATELLITE/STATST/produce\_rank\_histograms\_all\_locations.sh** is used for calculation of quantile space rank histograms and their summary statistics (Mean Square Deviation, MSD and p-value) for areas for the daily mean brightness temperature (btmp).

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **amsua\_stats.sh** script
2. Accessing raw data from model simulations, quantiles and bootstrap MSD statistics
3. Accessing file with list of areas and their geographical coordinates (i.e., centers of areas)
4. Compiling fortran program **rank\_histograms\_one\_location.f95** to calculate rank histograms
5. Initiating loop over all areas
6. Selecting the simulation data for area
7. Selecting the quantiles for area
8. Selecting the MSD for area
9. Calling python script **daymean.py** to produce daily mean value of brightness temperature
10. Producing the rank histogram for area
11. Converting the rank histogram file to ODB format
12. Combining the rank histogram data to all areas in a single ODB file

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable** – Channel number of AMSU-A satellite observations

*Note: currently only the channel number* ***5*** *is used*

*Note: currently only the brightness temperature, btmp (194), is evaluated*

1. **year1** – first year (for example, 2016)
2. **year2** – last year (for example, 2016)
3. **month1** – first month (for example, 01)
4. **month2** – last month (for example, 12)

**LIST OF INPUT DATA REQUIRED:**

1. file incl. list of locations for areas (it is assumed that the list of such locations has been pre-produced)

*Note: currently there is only 1 area called "a00" (with a size of about 100x100km over the Barents Sea) and which is assigned to exact geographical location (45 deg E & 70 deg N) i.e., in the center point of the area “a00”*

**obsall/SATELLITE/STATST/list\_of\_all\_amsua\_locations.txt**

1. path to location of raw simulation data for areas

**sim\_dir=/scratch/project\_465000454/ama/AMSUA\_demo\_ODB/test\_odb\_files**

1. path to location of file incl. quantiles as a function of time of year

**quant\_dir=/scratch/project\_465000454/ama/STATDATASATELLITE/quantiles**

1. path to location of file incl. rank histogram bootstrap MSD values

**bootstrap\_dir=/scratch/project\_465000454/ama/STATDATASATELLITE/bootstrap\_statistics**

**OUTPUT FOR RESULTS:**

1. path to store calculated output for each area in separate files

**outdir= obsall/SATELLITE/STATST/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2} outfile=${outdir}/rank\_histograms\_${variable}\_${station}\_${year1}${month1}-${year2}${month2}**

**FILE (FORTRAN PROGRAM): obsall/SATELLITE/STATST/rank\_histograms\_one\_location.f95** is called and compiled from **produce\_rank\_histograms\_all\_locations.sh** and calculates for each area the frequencies of quantiles having output incl. area code/ID, longitude, latitude, hour, MSD for the quantile frequencies, p-value of MSD (=fraction of bootstrap realizations with larger or equal MSD), and quantile frequencies (on scale 0-1) from 0-1% ... 99-100%.

**FILE (PYTHON PROGRAM): obsall/SATELLITE/STATST/python/daymean.py** calculates daily mean brightness temperature, btmp. Python program uses passed list of parameters: **hourly\_data\_file, daily\_data\_file** from the **produce\_rank\_histograms\_all\_locations.sh** bash-script.

### ***3.6.2. Standard plots for each area***

(with **produce\_standard\_plots\_all\_locations.sh** using **plot\_quantiles\_rankhist\_amsua.py** and **sql** with **odb\_api**; see an example of such plot in **Figure 3a**)

**FILE (BASH-SCRIPT): obsall/SATELLITE/STATST/produce\_standard\_plots\_all\_stations.sh** is used for accessing required data and calling for plotting quantiles rank histogram for each individual area.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **amsua\_stats.sh** script
2. Accessing raw data from model simulations, observations, quantiles, and calculated rank histogram data
3. Accessing file with list of areas and geographical coordinates of centres of these areas
4. Initiating loop over all areas
5. Selecting the simulation data for areas
6. Selecting the rank histogram data for areas
7. Selecting the quantiles for areas
8. Calling python script **plot\_quantiles\_rankhist\_amsua.py** to plot quantiles rank histogram for areas

**INPUT: LIST OF NECESSARY PASSED PARAMETERS:**

1. **variable** – Channel number of AMSU-A satellite observations

*Note: currently only the channel number* ***5*** *is used*

*Note: currently only the brightness temperature, btmp (194), is evaluated*

1. **year1** – first year (for example, 2016)
2. **year2** – last year (for example, 2016)
3. **month1** – first month (for example, 01)
4. **month2** – last month (for example, 12)

**LIST OF INPUT DATA REQUIRED:**

1. file incl. list of locations for areas (it is assumed that the list of such locations has been pre-produced)

*Note: currently there is only 1 area called "a00" (with a size of about 100x100km over the Barents Sea) and which is assigned to exact geographical location (45 deg E & 70 deg N) i.e., in the center point of the area “a00”*

**obsall/SATELLITE/STATST/list\_of\_all\_amsua\_locations.txt**

1. path to location of raw simulation data for areas

**sim\_dir=/scratch/project\_465000454/ama/AMSUA\_demo\_ODB/test\_odb\_files**

1. path to location of file incl. quantiles as a function of time of year

**quant\_dir=/scratch/project\_465000454/ama/STATDATASATELLITE/quantiles**

1. path to location of file incl. computed rank histogram data

**rh\_dir= obsall/SATELLITE/STATST/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**OUTPUT FOR RESULTS (PLOTS):**

1. path to store calculated output for each area in separate files

**figure\_dir=obsall/SATELLITE/STATST/figures/standard\_plots\_${variable}\_${year1}${month1}-${year2}${month2}**

**fig\_name=${figure\_dir}/standard\_plot\_${variable}\_${station}\_${year1}${month1}-${year2}${month2}.png**

*Note:* ***${station}*** *is read from file* ***list\_of\_all\_amsua\_locations.txt***

**FILE (PYTHON PROGRAM): obsall/SATELLITE/STATST/python/plot\_quantiles\_rankhist\_amsua.py** produces individual plot (see, for example, **Figure 3a** “Quantiles/ time series data and rank histogram for daily mean brightness temperature for areaID”) for each area. Python program uses passed list of parameters: **quantiles\_data, rank\_data, time\_series\_data, fig\_title, fig\_name, year\_beg, year\_end, month\_beg, month\_end, station\_id** from the **produce\_standard\_plots\_all\_locations.sh** bash-script.

A close-up of a graph

Description automatically generated

***Figure 3a:*** *Plots for brightness temperature /AMSU-A Channel 5/ for selected geographical area “a00” for period of 1 Jan - 31 Dec 2016: Quantiles/ time series data and rank histogram for daily mean brightness temperature for area “a00”.*

### ***3.6.3. Summary rank histograms for all areas***

(with **summary\_rank\_histograms\_all\_locations.sh** using **rank\_histogram\_sumstats\_locations.f95**, **plot\_p\_values\_map\_amsua.py**, **plot\_rankhist\_sum\_all\_areas\_amsua.py** and **sql** with **odb\_api**; see examples of such plots in **Figure 3b** and **Figure 3c**).

**FILE (BASH-SCRIPT): obsall/SATELLITE/STATST/summary\_rank\_histograms\_all\_locations.sh** is used for accessing required data, calculating rank histogram summary statistics over all areas, calling for plotting p-values over geographical domain for all areas, and calling for plotting rank histogram summary statistics based on all areas.

**PROCESSING STEPS**:

1. Passing list of necessary parameters from the **amsua\_stats.sh** script
2. Accessing rank histogram data
3. Compiling fortran program **rank\_histogram\_sumstats\_locations.f95** to calculate rank histogram summary statistics for daily mean brightness temperature for areas
4. Calling python script **plot\_p\_values\_map\_amsua.py** to plot p-values for daily man brightness temperature for all areas over geographical domain
5. Calling python script **plot\_rankhist\_sum\_all\_areas\_amsua.py** to plot rank histogram summary statistics for daily mean brightness temperature based on all areas

**INPUT: LIST OF NECESSARY PASSED PARAMETERS**:

1. **variable** – Channel number of AMSU-A satellite observations

*Note: currently only the channel number* ***5*** *is used*

*Note: currently only the brightness temperature, btmp (194), is evaluated*

1. **year1** – first year (for example, 2016)
2. **year2** – last year (for example, 2016)
3. **month1** – first month (for example, 01)
4. **month2** – last month (for example, 12)

**LIST OF INPUT DATA REQUIRED:**

1. path to location of file incl. computed rank histogram data

**rh\_dir= obsall/SATELLITE/STATST/rank\_histograms/${variable}\_${year1}${month1}-${year2}${month2}**

**OUTPUT FOR RESULTS (PLOTS):**

1. path to store plotted output (based on all areas) in separate files

**figure\_dir= obsall/SATELLITE/STATST/figures**

**figure\_dir/p-values\_map\_amsua\_${variable}\_${year1}${month1}-${year2}${month2}.png**

**figure\_dir/rankhist\_sumstats\_amsua\_${variable}\_${year1}${month1}-${year2}${month2}.png**

**FILE (PYTHON PROGRAM): obsall/SATELLITE/STATST/python/plot\_p\_values\_map\_amsua.py** produces a single plot (see, for example, ***Figure 3b*** “P-values for brightness temperature quantiles for areas”). Python program uses passed list of parameters: **p\_values\_file, p\_values** from the **summary\_rank\_histograms\_all\_locations.sh** bash-script.

**FILE (PYTHON PROGRAM): obsall/SATELLITE/STATST/python/plot\_rankhist\_sum\_all\_areas\_amsua.py** produces a single plot (see, for example, ***Figure 3c***“Rank histogram summary statistics for brightness temperature for areas: normalized p-value frequencies and normalized quantile frequencies”). Python program uses passed list of parameters: **p\_freq, q\_freq, number\_of\_stations, p01, p1, p5, max\_freq\_p, max\_freq\_q** from the **summary\_rank\_histograms\_all\_locations.sh** bash-script.

A map of the world

Description automatically generated

***Figure 3b:*** *Plots for brightness temperature /AMSU-A Channel 5/ for selected geographical area “a00” for period of 1 Jan - 31 Dec 2016: P-values for brightness temperature quantiles for areas. Note: on Figure 3b the plotted dot as a colored circle represents the central point (70 N, 45 E) of the area "a00", which is situated within boundaries of [69.5, 70.5] N and [43.5, 46.5] E.*

A graph of different values

Description automatically generated with medium confidence

***Figure 3c:*** *Plots for brightness temperature /AMSU-A Channel 5/ for selected geographical area “a00” for period of 1 Jan - 31 Dec 2016: Rank histogram summary statistics for brightness temperature for areas: normalized p-value frequencies and normalized quantile frequencies.*

# **CHAPTER 4: CONTRIBUTING TO OBSALL**

Contributions to the OBSALL Apps are always welcome! These inlude reporting on encountered bugs and issues, suggesting new features and enhancements, contributing to code and documentation. The sections below outline process for contributing to OBSALL.

## **Reporting bugs**

In case any bugs or issues are encountered while using OBSALL; please, follow steps below to report:

1. Navigate to the OBSALL GitLab repository (<https://earth.bsc.es/gitlab/digital-twins/de_340/obsall>).

2. Click on the “Issues” tab.

3. Click on the “New issue” button.

4. Choose to provide a title, type of issue, and detailed description of the issue, including steps to reproduce, expected behavior, and any relevant error messages & click on the “Create issue” button.

## **Suggesting features**

Any idea for new feature or enhancement in OBSALL is welcome; please, follow steps below to suggest:

1. Navigate to the OBSALL GitLab repository (<https://earth.bsc.es/gitlab/digital-twins/de_340/obsall>).

2. Click on the “Issues” tab.

3. Click on the “New issue” button.

4. Choose to provide a title, type of issue, and detailed description of the proposed feature, including use cases, benefits, and potential challenges & click on the “Create issue” button.

## **Contributing to code**

Contributions to the OBSALL code (SYNOP, TEMP, AMSU-A Parts and their monitoring/ statistics) are welcome; please, follow steps below to contribute:

1. Ask the OBSALL development team to grant you access to the OBSALL repository at GitLab.

2. Navigate to the OBSALL GitLab repository (<https://earth.bsc.es/gitlab/digital-twins/de_340/obsall>) & create an issue (following steps from 4.2) to discuss proposed changes/ features.

3. Clone the OBSALL repository to your local machine (e.g., git clone <https://earth.bsc.es/gitlab/digital-twins/de_340/obsall.git> & cd obsall)

4. Create a new branch for feature or bugfix (e.g., git checkout -b my-branch).

5. Make and commit your changes (git status & git add file-name-with-changes-made & git commit -m "write-in-short-about-changes-made”)

6. When ready, push your branch to the OBSALL repository (git push origin my-branch)

7. Create a pull request in the OBSALL repository, describing your changes and referencing any related issues.

8. Once your pull request is approved, it will be merged into the main branch by the OBSALL development team.

**Warning**: Please, do not merge your pull requests into the OBSALL main branch by yourself & do not commit (!never!) any changes directly to the main branch!

## **Contributing to documentation**

Any suggestions on improvements to the OBSALL documentation are always welcome; please, follow same steps as explained in 4.2.

# **CHAPTER 5: TROUBLESHOOTING AND FAQ**

Frequently asked questions and addressed common issues (that users may encounter when working with OBSALL) are given below.

**Q**: I am getting an error when trying to clone OBSALL on Lumi, what can I do?

**A**: Check that you have access to the OBSALL repository at GitLab & check that you typed login and password correctly when accessing repository.

**Q**: I am getting an error (as an infinite looping) when trying to load all required modules for OBSALL on Lumi, what can I do?

**A**: Interrupt such loop, logout from your account, login again to your account on Lumi & repeat all previous steps again.

**Q**: How do I cite OBSALL in my research or publications?

**A**: When using OBSALL for research purposes, please, cite it and its authors using information provided in the Chapter “REFERENCES AND ACKNOWLEDGMENTS”.

# **CHAPTER 6: REFERENCES AND ACKNOWLEDGMENTS**

## **Citing OBSALL**

If you use OBSALL Apps your research or publications, please, cite the Apps using the following format and please include also the following acknowledgements:

@software{obsall,

author = {Heikki Järvinen, Jouni Räisänen, Lauri Tuppi, Madeleine Ekblom, Alexander Mahura},

title = {ObsAll: Processing ground-based, radiosounding and satellite observational data for streamed climate modeled data},

year = {2024},

publisher = {GitLab},

journal = {Barcelona Supercomputing Center, Earth Sciences GitLab},

howpublished = {\url{https://earth.bsc.es/gitlab/digital-twins/de\_340/obsall/newbranch}},

}

## **Acknowledgments**

The authors would like to thank many colleagues who have helped in developing, testing, implementing, and maintaining the OBSALL Apps.

The development of OBSALL has been funded by the European Union (EU) through Contract DE\_340\_CSC – Destination Earth Programme Climate Adaptation Digital Twin (Climate DT). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of EU, the European Commission (EC) or ECMWF. Neither EU and EC nor ECMWF can be held responsible for them.

The Destination Earth is EU funded initiative, and it is implemented by ECMWF (European Centre for Medium-Range Weather Forecasts), ESA (European Space Agency), and EUMETSAT (European Organization for the Exploitation of Meteorological Satellites).

## **Especial thanks**

Our especial thanks:

* for advise on issues with autosubmit, gsv extraction and other tools/apps, integration in workflow, etc. to BSC colleagues: *Miguel Castrillo, Francesc Roura, Iker Gonzalez, Marvin Axness, Aina Gaya i Àvila*;
* for constructive comments, remarks, suggestions to reviwers: *Steger Christian (DWD), Bruno Kinoshita (BSC), Miguel Castrillo (BSC) & Aina Gaya i Àvila (BSC), Pablo Ortega (BSC), Paolo Davini (CNR), Jost Hardenberg (POLITO), Natalia Nazarova (POLITO)*;
* for Lumi related advises/help to: *Juha Lento (CSC), Devaraju Narayanappa (CSC), Henrik Nortamo (CSC), Sami Salonen (CSC), Roman Nuterman (UCPH), Thomas Geenen (ECMWF)*; and
* for polytope tool info/examples sharing to: *Matthew Griffith, Nils Wedi (ECMWF)*.