Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

# R tools user meeting

An-Chi Ho and Núria Pérez-Zanón

contributors: Jan Mateu

09/04/2021

# Agenda

1. Package update
   - CDO
   - startR
2. What are CST functions in CSTools and CSIndicators packages? [Núria]
3. CALIOPE-Urban: from non-uniform mesh to regular latlon grid (Jan)
4. Q&A

# CDO version

The functions that use CDO internally, e.g., s2dv::CDORemap, could be sensitive to the CDO module version. Operation may fail simply due to the version.

It is recommended to use **R/3.6.1-foss-2015a-bare** and **CDO/1.9.8-foss-2015a** in principle. You can add them in .bashrc, e.g.,

```
alias rdev='module load CDO/1.9.8-foss-2015a R/3.6.1-foss-2015a-bare'
```

# startR

# multiStart()

The wrapper of Start():

- Retrieve multiple datasets that have different calendars or lead time lengths together.
- Assign different selector indices to each dataset, including NA.
- Use “list of list” to define the dimensions that are different among datasets.

```
data <- multiStart(
        dat = list(list(name = 'hadgem3', path = path_hadgem3),
                   list(name = 'mpi_esm', path = path_mpi_esm)),
        var = 'tasmax',
        sdate = '2000',
        fyear = list(list(name = 'hadgem3', fyear = fyear_hadgem3),
                     list(name = 'mpi_esm', fyear = fyear_mpi_esm)),
        time = list(list(name = 'hadgem3', time = time_hadgem3),
                    list(name = 'mpi_esm', time = time_mpi_esm)),
        lat = 'all',
        lon = 'all',
        retrieve = TRUE)
```

common among datasets

different among datasets

```
dim(data)
  dat   var sdate fyear  time   lat   lon
    2     1     1     1     3    61    51
```

# multiStart()

The wrapper of Start():

- If retrieve = FALSE, the datasets are not combined together actually. They're saved separately in one object.
- Therefore, the function and workflow should be defined as for two datasets rather than one.

```
str(data)
#List of 2
# $ hadgem3: language Start ...
#  ..- attr(*, "Dimensions") ...
#  ..
#  ..- attr(*, "Variables") ...
#  ..
# $ mpi_esm: language Start ...
#  ..- attr(*, "Dimensions") ...
#  ..
#  ..- attr(*, "Variables") ...
```

```
func_sum <- function(x, y) {
  return(x + y)
}

step <- Step(func_sum,
             target_dims = list(hadgem3 = c('dat'),
                                mpi_esm = c('dat')),
             output_dims = c('dat'))

wf <- AddStep(data, step)

output <- Compute(wf, chunks = list(time = 2))
```

# multiStart()

**[Example 1]**
Two datasets with different calendars. NAs need to be inserted in the time dimension so the time steps between datasets can be aligned.
https://earth.bsc.es/gitlab/es/startR/-/blob/develop-multiStart/inst/doc/usecase/ex2_11_multiStart_insert_NA.R

**[Example 2]**
Three datasets with different initial dates. NAs need to be inserted.
- HadGEM3 (initialised in November)
- IPSL-CM6A-LR (initialised in January)
- NorCPM1 (initialised in October)

https://earth.bsc.es/gitlab/es/startR/-/blob/develop-multiStart/inst/doc/usecase/ex2_12_multiStart_diff_init_dates.R

**[Example 3]**
Read EC-Earth dataset (several files for one sdate) together with other datasets (one file for one sdate).
https://earth.bsc.es/gitlab/es/startR/-/blob/develop-multiStart/inst/doc/usecase/ex1_14_multiStart_ecearth_merge.R



Try to use multiStart() with the branch "develop-multiStart".

# What is a CST function in CSTools and CSIndicators packages?

# CST functions

**Aim:** To clarify the difference between functions **with and without prefix CST** on CSTools and CSIndicators packages to facilitate their usage and development.

The documentation of these packages shows functions exposed to users. E.g.:

| | with prefix | without prefix | |
|---|---|---|---|
| **Downscaling** | | | |
| RainFARM: | CST_RainFARM | RainFARM | vignette |
| Analogs: | CST_Analogs | Analogs | - |
| **Forecast correction:** | | | |
| Best Estimate Index | CST_BEI_Weighting | BEI_Weightihg | vignette |
| Calibration methods: | CST_Calibration | Calibration | - |
| Bias Correction: | CST_BiasCorrection | BiasCorrection | - |
| Quantile Mapping: | CST_QuantileMapping | QuantileMapping | - |
| **Ensemble Clustering:** | CST_EnsClustering | EnsClustering | vignette |
| | **s2dv_cube objects** | **array objects** | |

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

# CST functions

**A) When reading NetCDF files, the information requested is retrieve on the R session:**

- **data**: the simulated 2m air temperature in mediterranean region (-20 to 51 ºE - 25 to 57 ºN) for the next day using the latest initialized simulation

**B) In some cases, more info may be needed (e.g. for plotting):**

- **lat**: the corresponding latitudes for the centers of the grid cells for the mediterranean region retrieved in the model grid
- **lon**: the corresponding longitudes for the centers of the grid cells for the mediterranean region retrieved in the model grid
- **time**: the corresponding dates (and hour) of the simulated time steps

**C) Other relevant information (e.g. for reproducibility):**

- **units, calendar, projection**
- **model simulation name**
- **files name and path**
- **date of execution**

*useful information in case there are changes on the archive or the package version

# CST functions

```
library(ncdf4)
file <-
nc_open('/esarchive/recon/ecmwf/era5/monthly_mean/sfcWind_f1h/sfcWind_201811.nc'
)
data <- ncvar_get(file, 'sfcWind', start = c(1, 1, 1), count = c(10, 10, 1))
lon <- ncvar_get(file, "lon")
lat <- ncvar_get(file, "lat", verbose = F)
t <- ncvar_get(file, "time")
tunits <- ncatt_get(file, "time", "units")
calendar <- ncatt_get(file, "time", "calendar")
time <- as.Date(...)
dunits <- ncatt_get(file, "sfcWind", "units")
grid <- ncatt_get(file, "sfcWind", "grid_type")
execution_date <- Sys.time()
nc_close(file)

> ls()
 [1] "calendar"       "data"           "dunits"         "execution_date"
 [5] "file"           "grid"           "lat"            "lon"
 [9] "proj"           "t"              "tunits"         "dates"
```

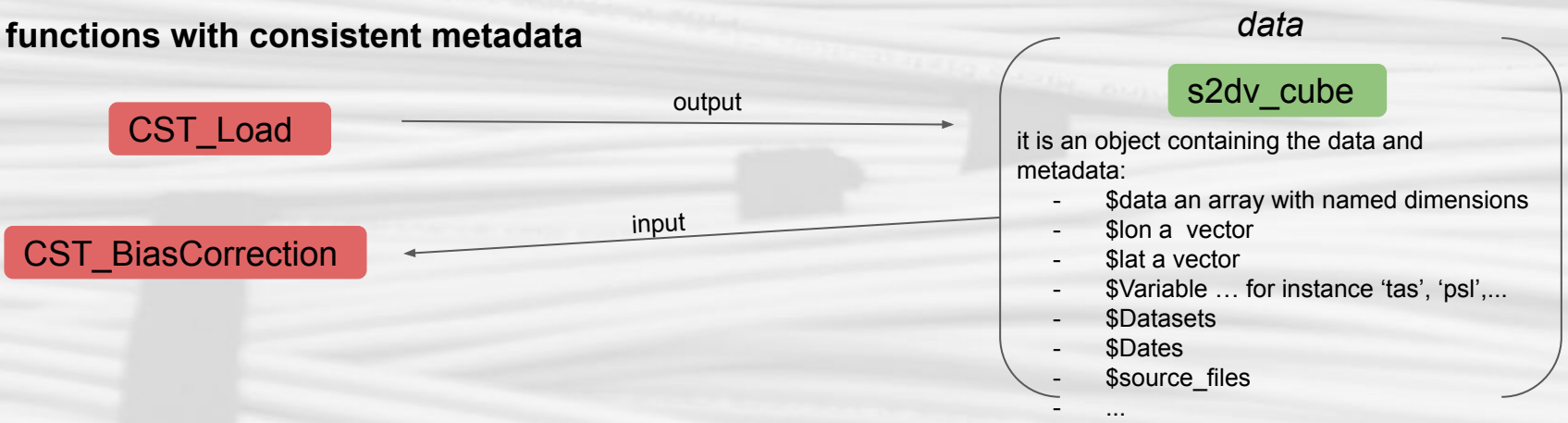
# CST functions

```
library(CSTools)
library(zeallot)
start <- as.Date(paste(1993, '05' , "01", sep = ""), "%Y%m%d")
end <- as.Date(paste(1996, '05', "01", sep = ""), "%Y%m%d")
dateseq <- as.character(format(seq(start, end, by = "year"), "%Y%m%d"))
c(exp, obs) %<-%  CST_Load(var = 'tas',
                  exp = 'system5c3s',
                  obs = 'erainterim',
                  sdates = dateseq, leadtimemin = 1, leadtimemax = 1,
                  lonmin = -19, lonmax = 60.5, latmin = 0, latmax = 79.5,
                  storefreq = "daily", sampleperiod = 1,  nmember = 5,
                  output = "lonlat", method = "bilinear",
                  grid = "r360x180")

> names(exp)
[1] "data"             "lon"              "lat"              "Variable"
[5] "Datasets"         "Dates"            "when"             "source_files"
[9] "load_parameters"
> class(exp)
[1] "s2dv_cube"
```

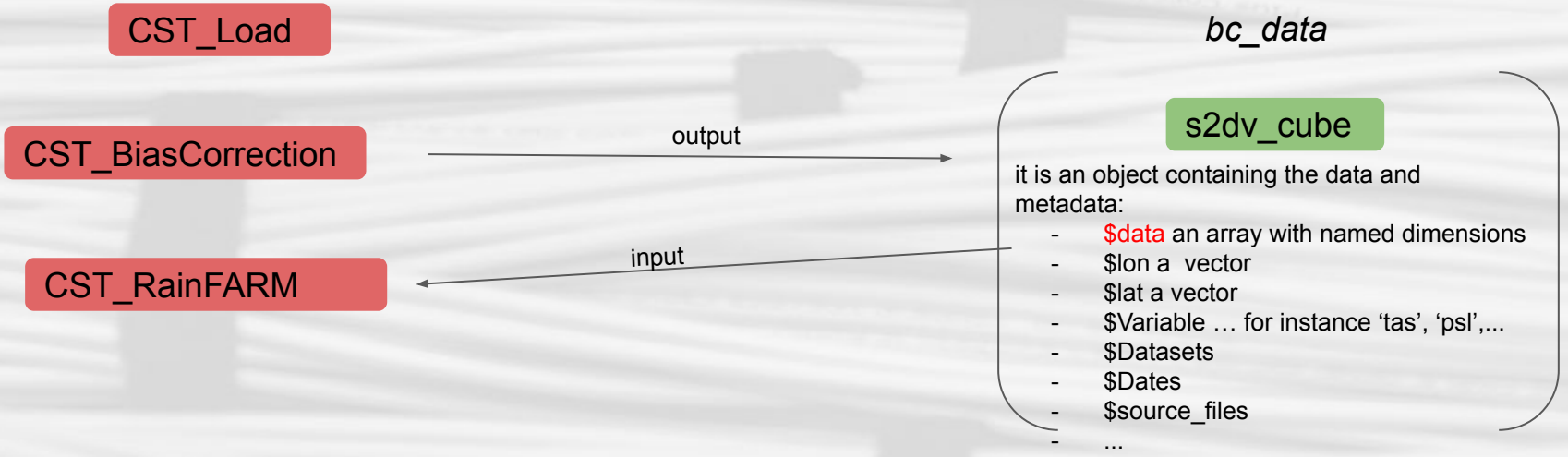
# CST functions

**Chain of functions with consistent metadata**



# CST functions

**Chain of functions with consistent metadata**



# CST functions

**Chain of functions with consistent metadata**

CST_Load

CST_BiasCorrection

CST_RainFARM

CST_PeriodAccumulation

output

input

*ds_bc_data*

s2dv_cube

it is an object containing the data and metadata:

- $data an array with named dimensions
- $lon a vector
- $lat a vector
- $Variable … for instance ‘tas’, ‘psl’,...
- $Datasets
- $Dates
- $source_files
- ...

# CST functions

**Chain of functions with consistent metadata**

CST_Load

CST_BiasCorrection

CST_RainFARM

CST_PeriodAccumulation

output

*sectorial indicator*

s2dv_cube

it is an object containing the data and metadata:

- $data an array with named dimensions
- $lon a vector
- $lat a vector
- $Variable … for instance ‘tas’, ‘psl’,...
- $Datasets
- $Dates
- $source_files
- ...

*The order and number of steps on the chain of functions is not fix.

# CST functions

CST_**Method**(data1, data2, method = 'method1', ...) {

**Method**(data1$data, data2$data, data1$Dates, data2$Dates...) {

**multiApply::Apply**(list(data1$data, data2$data),
target_dims = ….
fun = **.method**)

}

**.method** -> function(vector1, vector2,...)

*data1 and data 2 are s2dv_cubes*

s2dv_cube

it is an object containing the data and metadata:

- $data an array with named dimensions
- $lon a vector
- $lat a vector
- $Variable … for instance 'tas', 'psl',...
- $Datasets
- $Dates
- $source_files
- ...

- **.method** function inputs have the minimum required dimensions to define the calculation
- **multiApply::Apply** takes care of applying the fundamental function in the rest of the dimensions as many times as necessary and it can be done in parallel.

# CST functions

- **s2dv_cube's** can be created with **CST_Load()**, **as.s2dv_cube()** or **s2dv_cube()**
- It is not mandatory to use **CST_** functions.

➔ **CSTools** and **CSIndicators** include functions with prefix **CST_**

➔ **CSTools**, **CSIndicators**, **ClimProjDiags** and **s2dv** include function without prefix (the Apply function) and they are build under similar guidelines (i.e. they request **N-Dimensional arrays with named dimensions** as inputs)

- When developing a function, the developer begins (1) writing the most inner function (**.method**), then can (2) wrapper it with **Apply**, and check arguments and finally, create the CST_ function working with (3) s2dv_cube objects. Guidelines document available.

- The functions without prefix can easily being used in the startR chunking workflow.
- The functions with prefix are easier to be used by end-users from that want to create their own applications.

# CALIOPE-Urban

## From a non-uniform mesh to a regular one

J.M. Armengol, N. Pérez-Zanón, A. Criado, J. Benavides, A. Soret, H. Petetin,

O. Jorba, M. Guevara , D. Rodríguez, C. Pérez García-Pando,

F. Macchina, C. Tena, M. Olid, G. Montane, K. Serradell

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

# CALIOPE-Urban

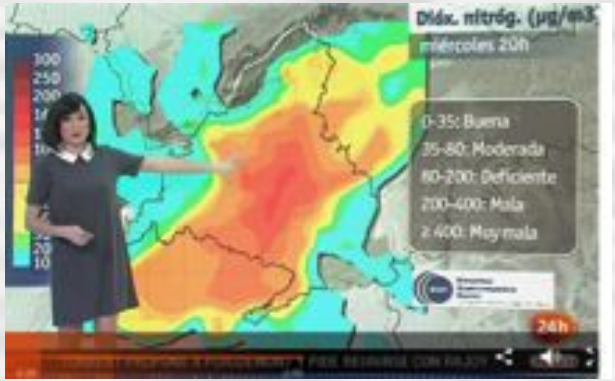
## Agenda

- Intro to CALIOPE-Urban
- Re-gridding with gstat
- Future works

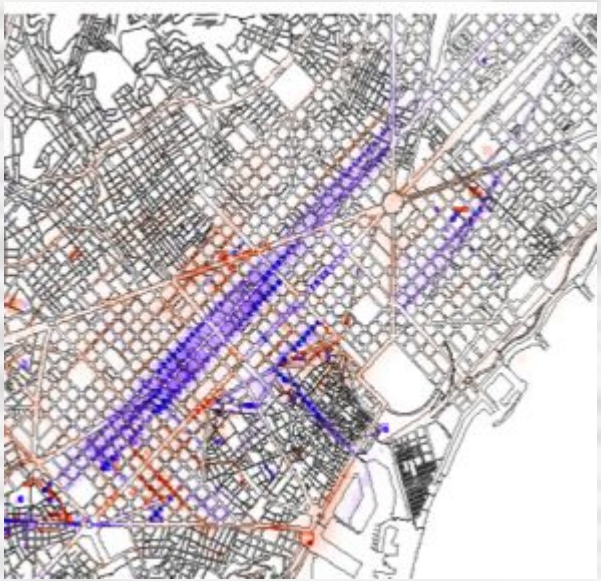
# CALIOPE-Urban

Air quality models

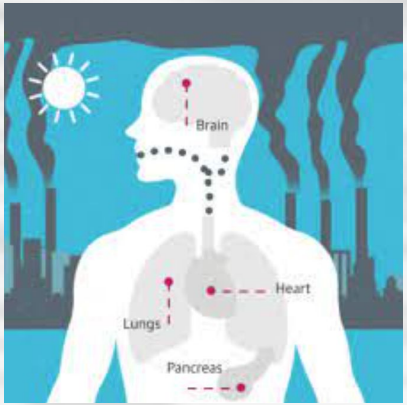
Forecasting




Scenarios testing



Heath impact studies

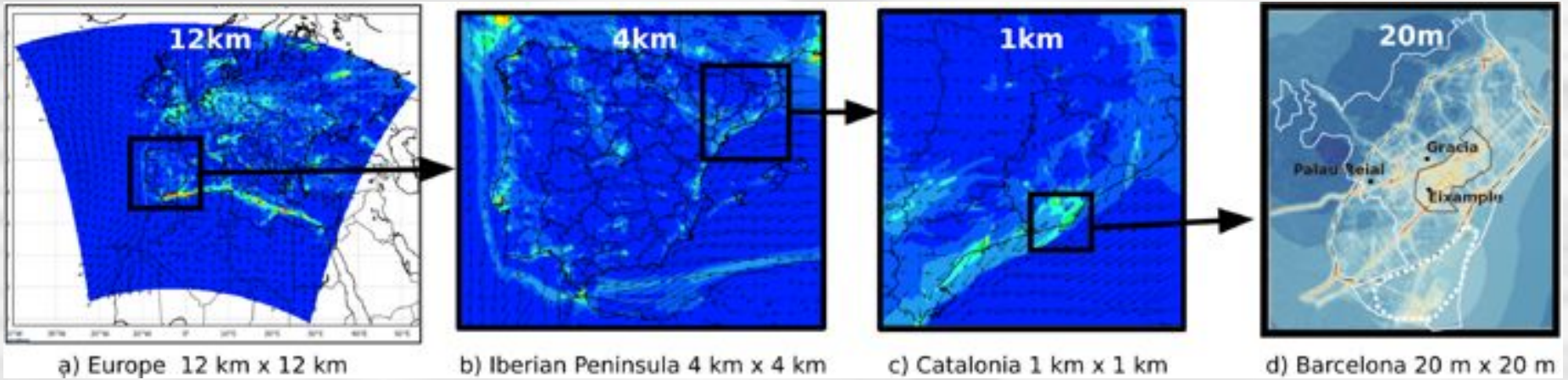


# CALIOPE-Urban

Air quality models

Regional model

Urban model




a) Europe 12 km x 12 km

b) Iberian Peninsula 4 km x 4 km

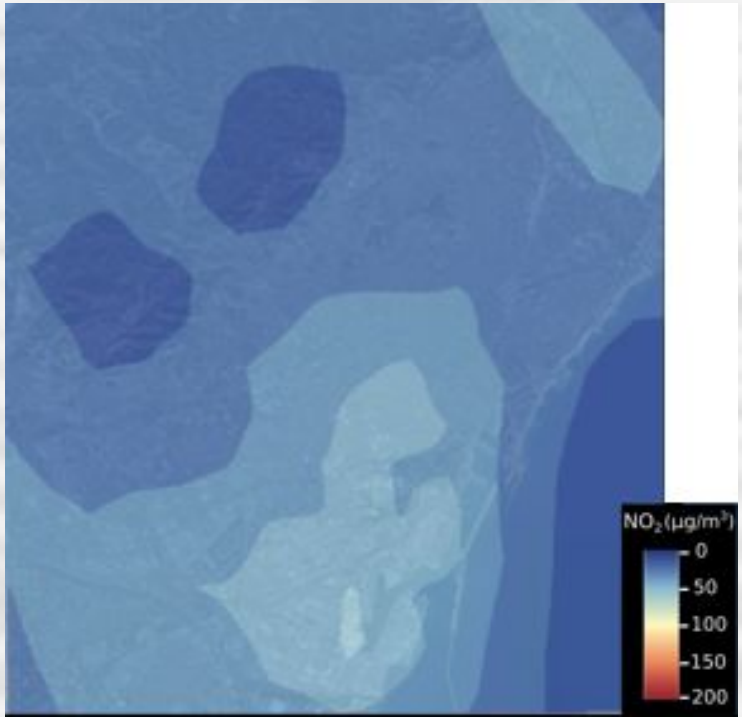
c) Catalonia 1 km x 1 km

d) Barcelona 20 m x 20 m

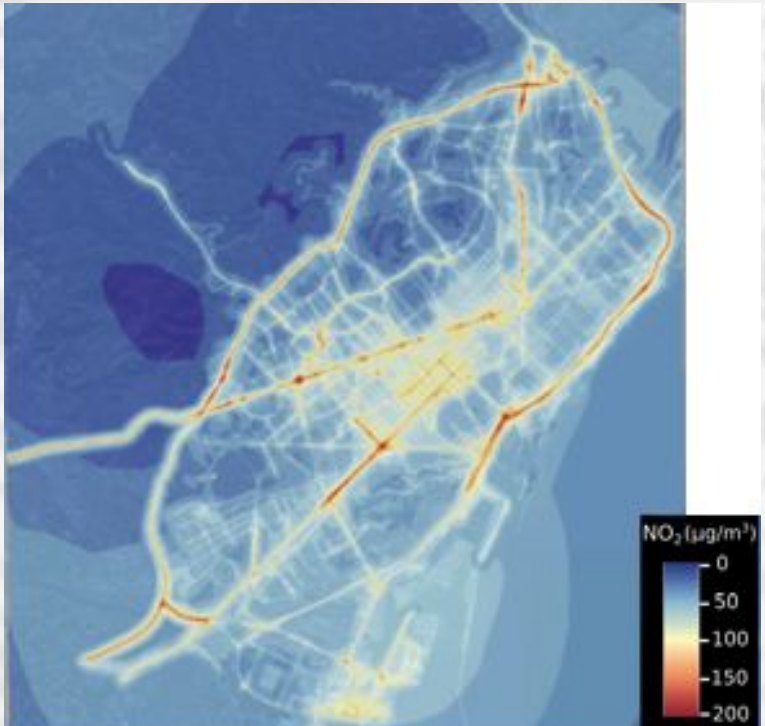
# CALIOPE-Urban

Need for more resolution in the cities

CALIOPE 1kmx1km resolution

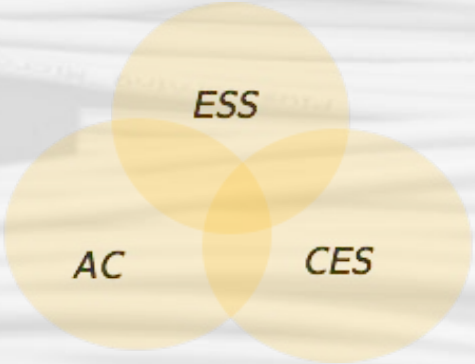



CALIOPE–Urban 20mx20m resolution

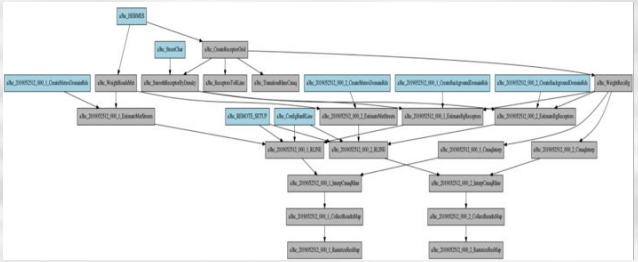


# CALIOPE-Urban

Ongoing: auto-CALIOPE-Urban operational




- autosubmit



- CPU time optimization

# CALIOPE-Urban

Uniform mesh

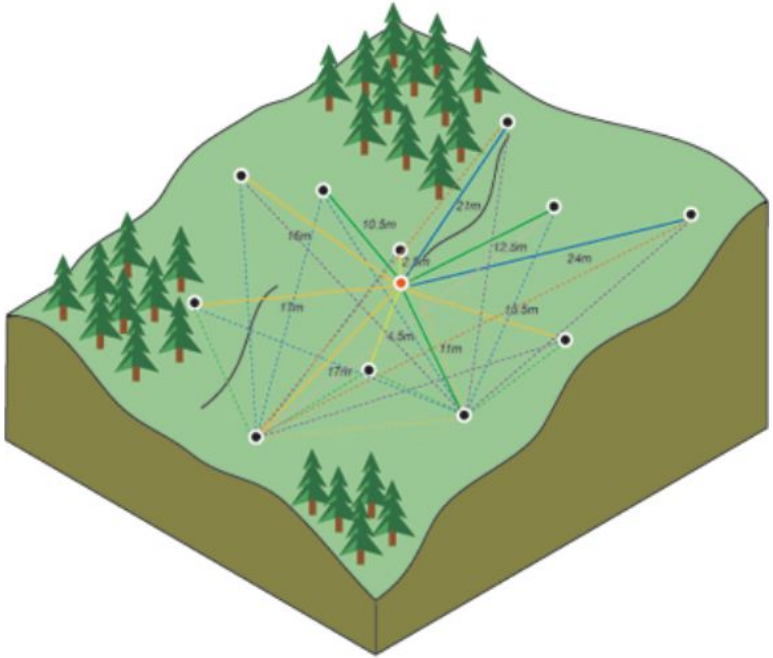

# CALIOPE-Urban

Non-Uniform mesh

482 receptors, 1st layer 500x500m beyond 300m
2nd layer 75x75m between 40-300m
3rd layer 20x20m less than 40m

# CALIOPE-Urban

Re-gridding: 1st option IDW


$$\hat{Z}(s_0) = \frac{\sum_{i=1}^{n} w(s_i) Z(s_i)}{\sum_{i=1}^{n} w(s_i)} \qquad w(s_i) = \frac{1}{d(s_0, s_i)^p}$$

# CALIOPE-Urban

## Re-gridding

```
# Load data:
results <- readRDS(paste0(output_path, "bcn_NonUniformGrid.rds"))
receptor <- readRDS(paste0(output_path, "target_uniform_grid_bcn.rds"))
dates <- unique(results$date)

# Re-gridding
        index <- array(1:length(dates), c(pos_index = length(dates)))
        res <- multiApply::Apply( list(index),
                                  fun = regrid,
                                  non_uniform_data = results,
                                  target_grid = receptor,
                                  date = dates, margins = 'pos_index',
                                  ncores = 16)

# Save NetCDF file:
...
```

```
regrid <- function(index, non_uniform_data, target_grid, date,
                   crs_sim = "+proj=utm +zone=31 +ellps=intl +units=m +no_defs") {

  # data.frame and SpatialPointsDataFrame:
        non_uniform_data <- results[results$date == date[index],]
        non_uniform_data <- sp::SpatialPointsDataFrame(non_uniform_data[,c("X", "Y")],
                                                       non_uniform_data)
        sp::proj4string(non_uniform_data) <- crs_sim

  # Re-gridding
        regular <- gstat::idw(NO2 ~ 1, non_uniform_data, newdata = target_grid,
                              idp = 2.0, nmax = 4)

  # Transform to lat-lon crs
        regular <- sp::spTransform(regular, sp::CRS("+proj=longlat"))
        lon <- unique(round(regular@coords[,1],7))
        lat <- unique(round(regular@coords[,2],7))
        proj <- paste(regular@proj4string@projargs, "+no_defs")
        regular <- regular@data$var1.pred
        dim(regular) <- c(lon = 1450, lat = 1644)

        return(list(regular = regular, lat = lat, lon = lon))
}
```

**the full code can be found in:**
**/esarchive/scratch/nperez/git/Flor/Jan4Nuria/test_regrid_nord3**

# CALIOPE-Urban

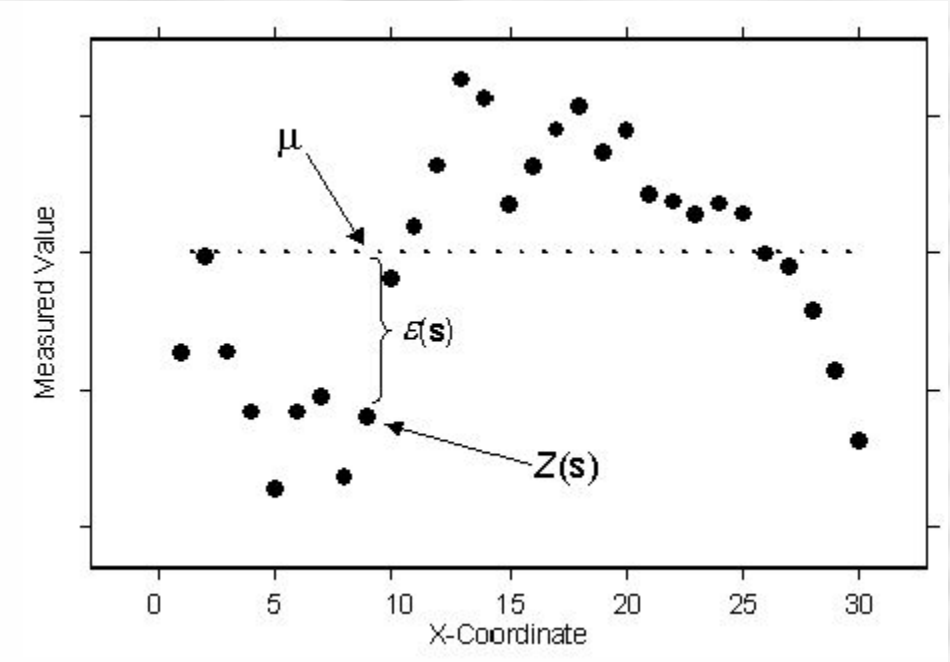
CPU time

| RLINE_street_gfort.x | Uniform mesh (20m x 20m) | Non-Uniform mesh (20m x 20m) + (75m x 75m) + (500m x500m) |
|---|---|---|
| Number of receptors | 277 884 nodes | 130 442 nodes |
| CPU time | 35 min (96 procs) | 18 min (96 procs) |

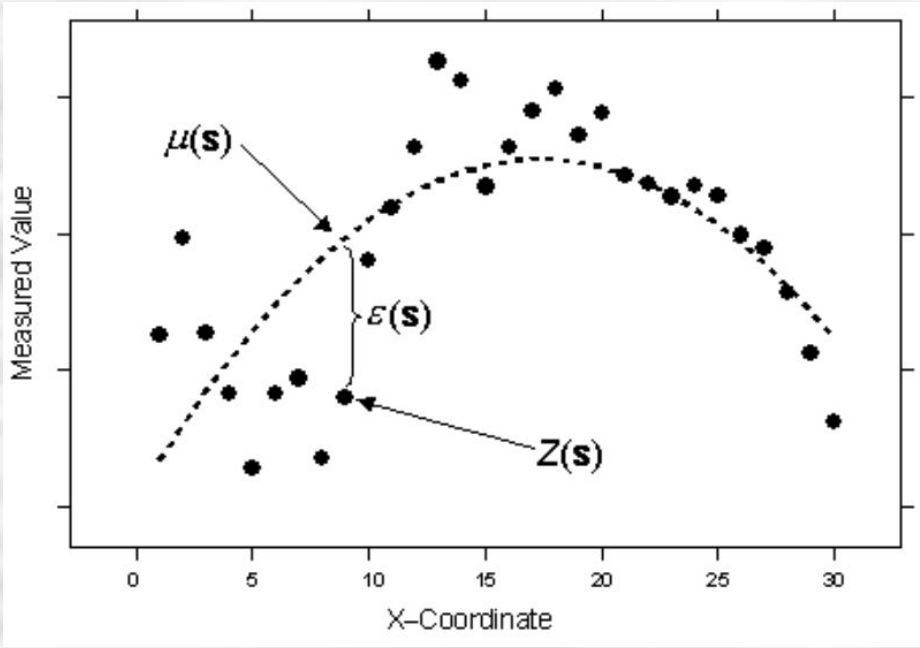
# CALIOPE-Urban

Future works: explore gstat capabilities

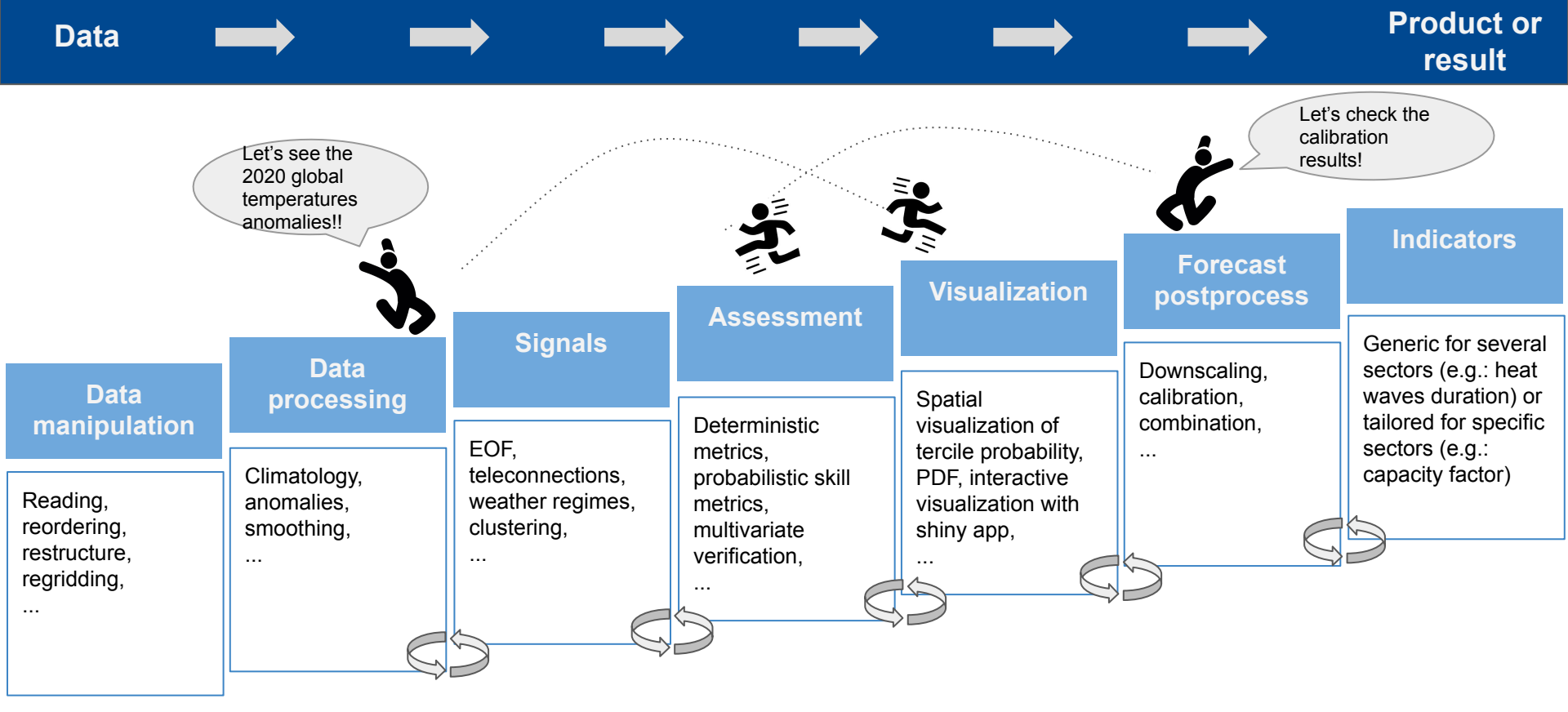
Ordinary Kriging $Z(\mathbf{s}) = \mu + \varepsilon(\mathbf{s}),$




Universal Kriging $Z(\mathbf{s}) = \mu(\mathbf{s}) + \varepsilon(\mathbf{s}),$



# Slides for the ‘Department Day’ meeting


Data
Product or result
Let's see the 2020 global temperatures anomalies!!
Let's check the calibration results!
Data manipulation
Reading, reordering, restructure, regridding, ...
Data processing
Climatology, anomalies, smoothing, ...
Signals
EOF, teleconnections, weather regimes, clustering, ...
Assessment
Deterministic metrics, probabilistic skill metrics, multivariate verification, ...
Visualization
Spatial visualization of tercile probability, PDF, interactive visualization with shiny app, ...
Forecast postprocess
Downscaling, calibration, combination, ...
Indicators
Generic for several sectors (e.g.: heat waves duration) or tailored for specific sectors (e.g.: capacity factor)

ClimProjDiags
s2dv/s2dverification
CSIndicators
startR
CSTools
easyNCDF
multiApply

★ **R tools are being used in several research lines and operationals**


| Research line | Projects | Publication e.g. |
|---|---|---|
| **In-situ observations** | Indecís, S2S4E | Tall towers and reanalysis Ramon et al. 2019 |
| **Atmospheric Composition** | Ongoing collaboration in CALIOPE-Urban | |
| **Sub-seasonal Forecast** | S2S4E | Verification Manrique et al. 2020 |
| **Seasonal Forecast** | S2S4E, Visca, Medscope, Medgold, QA4Seas | Wind power generation Lledó et al., 2019 |
| **Decadal Predictions** | EUCP, C3S 34c | CMIP6 Assessment Bilbao et al. 2021 |
| **Climate Projections** | C3S MAGIC | ESMValTool papers: python and R synergy |

# Q & A

Next meeting: 7th May 2021 (Friday 3pm)