Barcelona Supercomputing Center
Centro Nacional de Supercomputación

startR: A tool for large multi-dimensional data processing

Núria Pérez-Zanón*, An-Chi Ho*, Nicolau Manubens*, Francesco Benincasa*, Pierre-Antoine Bretonnière*

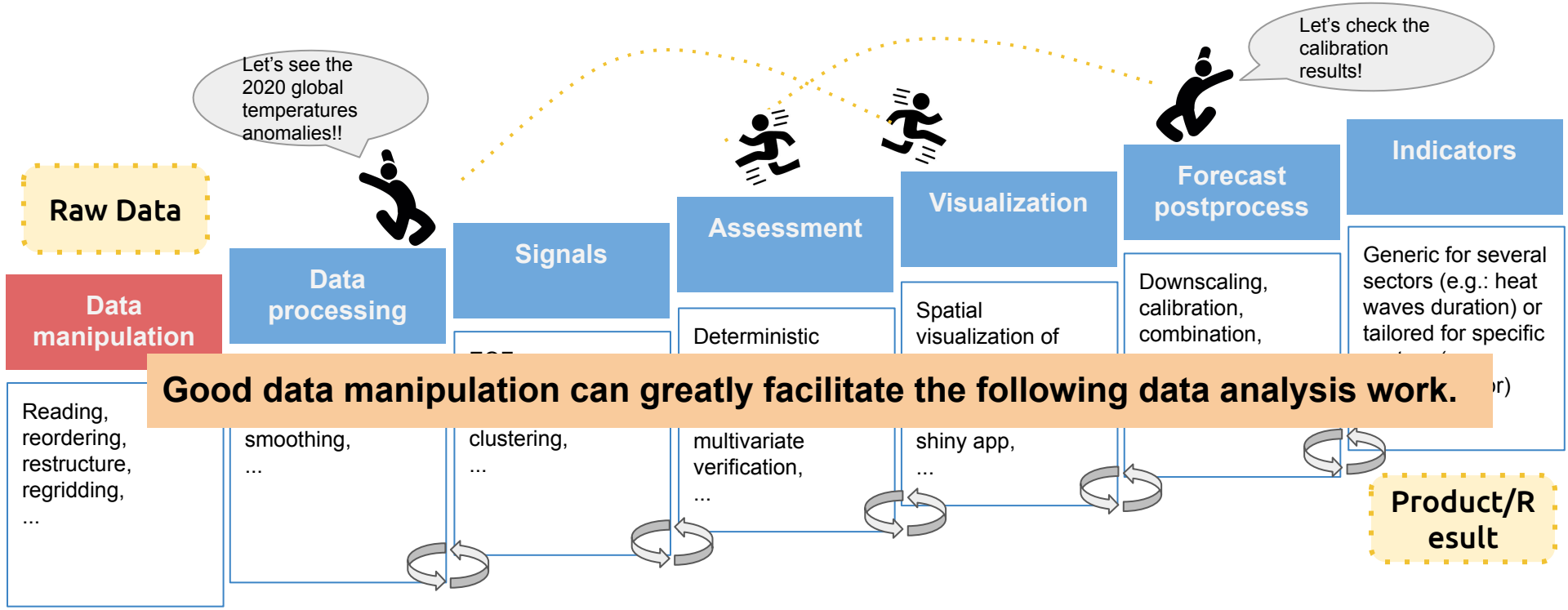
**Earth Science Department, Barcelona Supercomputing Center, Barcelona, Spain*

8th BSC Doctoral Symposium (2021)

Outline

1. startR's background and purpose
2. Functions and workflow
3. Use case
4. Summary and resources

Research process and R tools in Earth Science field



Good data manipulation can greatly facilitate the following data analysis work.

ClimProjDiags

CSIndicators

s2dv/s2dverification

startR

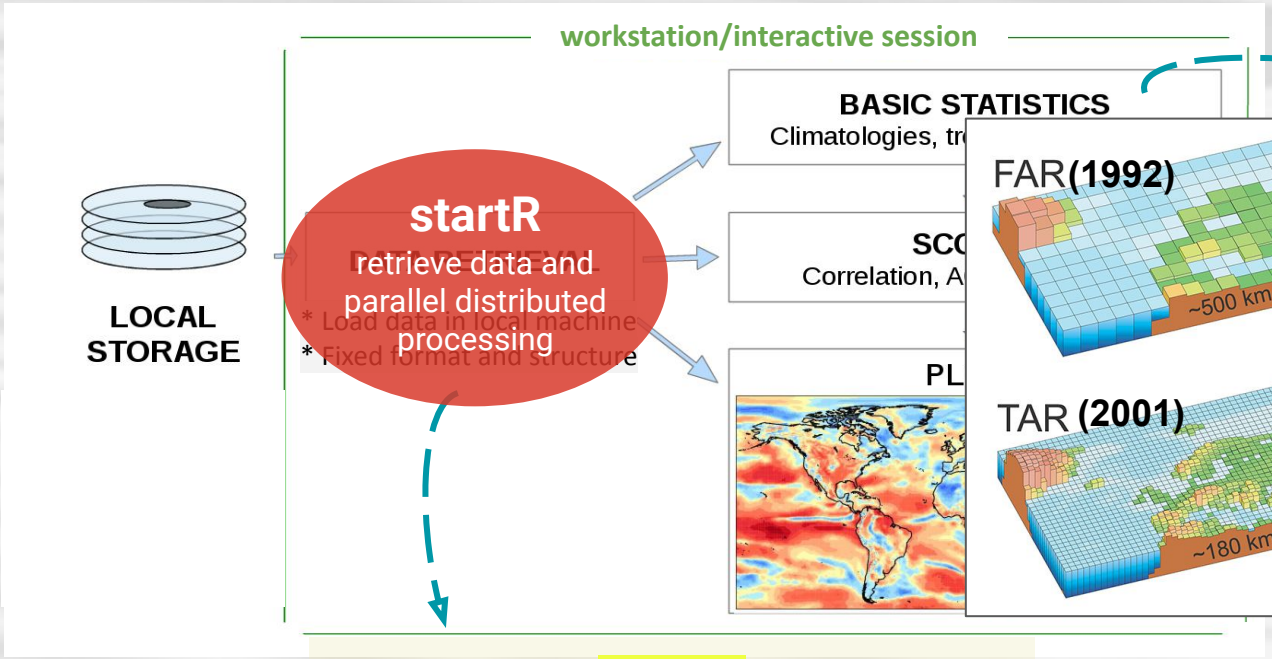
CSTools

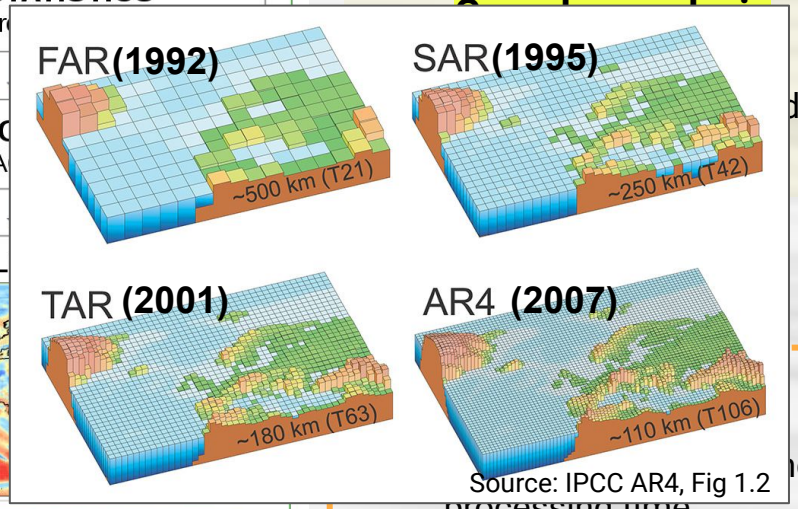
easyNCDF

multiApply

How was startR born?

[Data analysis procedure]





Source: IPCC AR4, Fig 1.2

Big data

- Higher resolution in all dimensions
- Various types of data required

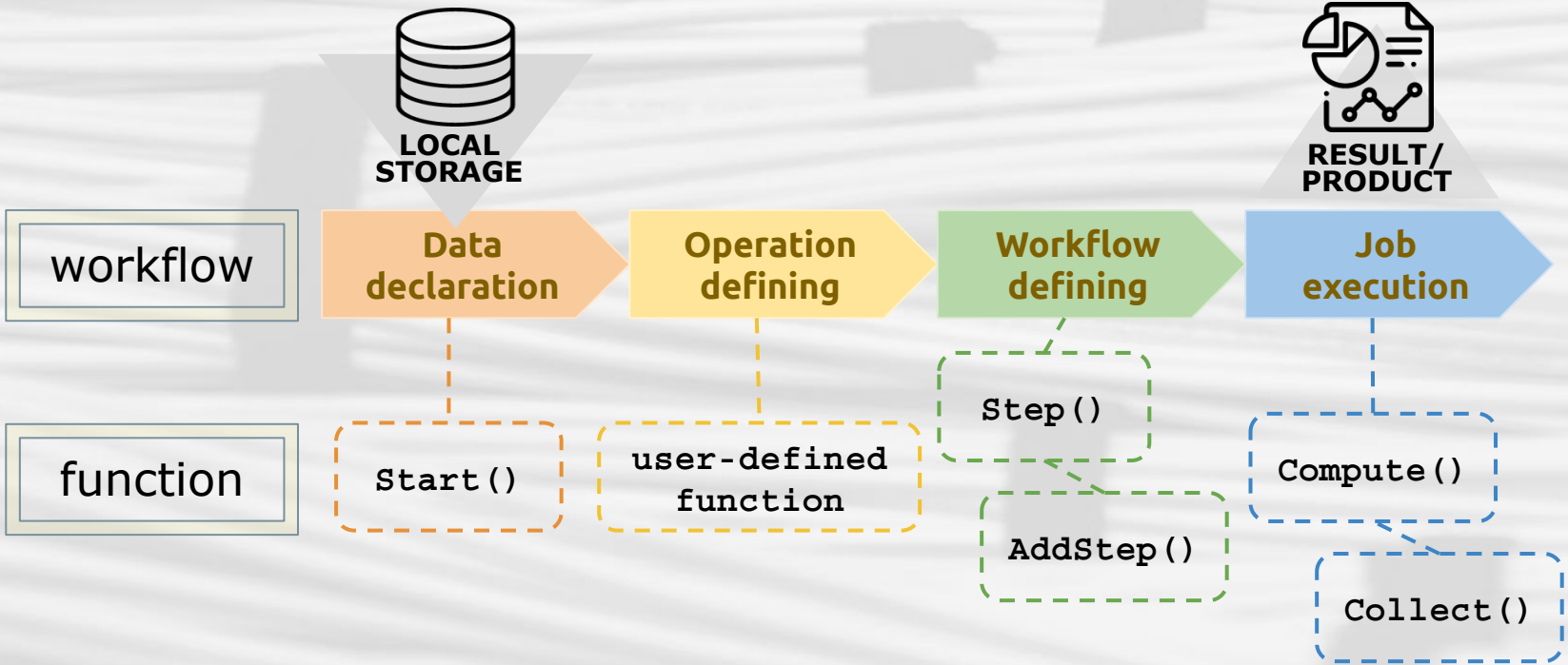
- Limited memory space in local machine

startR features

- ★ An R package tailored for big multi-dimensional data retrieval and processing
- ★ Apply multiApply paradigm, which provides flexibility in multi-dimensional data processing
- ★ Implement the MapReduce paradigm (i.e., chunking) on HPCs for parallel distributed data-processing
- ★ Pre-processing: data transformation or reordering/reshaping/renaming dimensions before performing analysis
- ★ Well-preserved metadata during the whole process
- ★ Use ecFlow workflow manager for job distribution and monitoring on HPCs
- ★ Acceptable data format: netCDF for now, but may be available for other formats.

startR functions and workflow

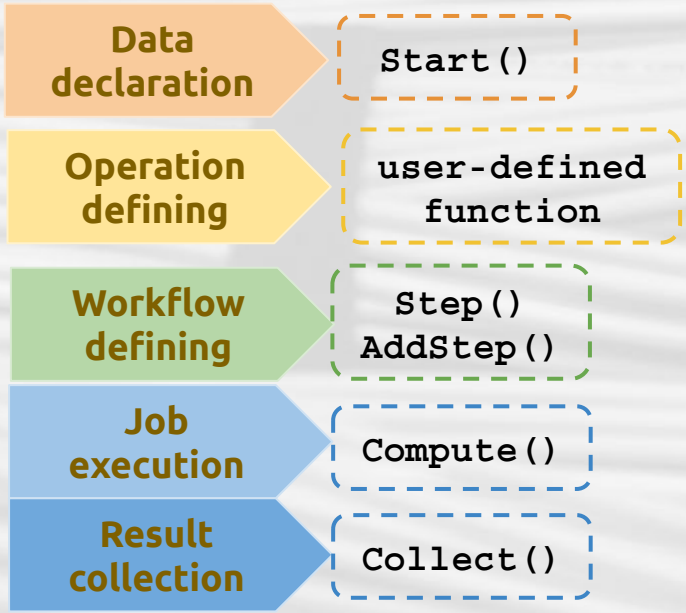
With startR, users can create a concise script for data analysis with all the information needed.



And other helper functions.

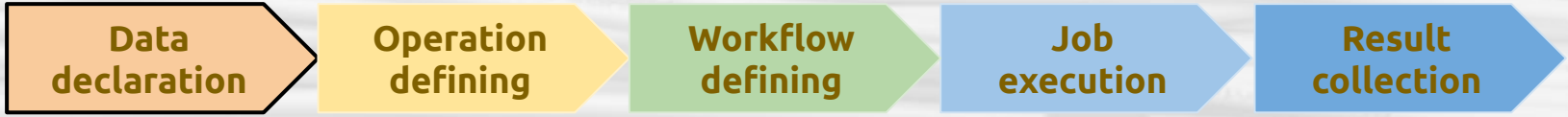
startR functions and workflow

With startR, users can create a concise script for data analysis with all the information needed.



1. Declare the data sources and the required file/inner dimensions.
2. Define the operations to be applied.
3. Combine the elements from the previous steps to build up the workflow.
4. Set the configuration for the chosen machine and trigger job execution.
5. Collect the results when the execution is finished.

1. Data declaration



- Identify which and where the files are and define the dimensions required.
- `Start()` will return a multi-dimensional array or an object with startR_cube class.

```
repos <-
'/esarchive/exp/ecmwf/system5_m1/monthly_mean$var$_f6h/$var$_$sda
te$.nc'

data <- Start(dat = repos,
              var = 'tas',
              sdate =  c('20170101', '20170201'),
              ensemble = indices(1:50),
              time = 'all',
              latitude = values(list(lat.min, lat.max)),
              longitude = values(list(lon.min, lon.max)),
              ...,
              retrieve = FALSE)
```

data source

file dimension

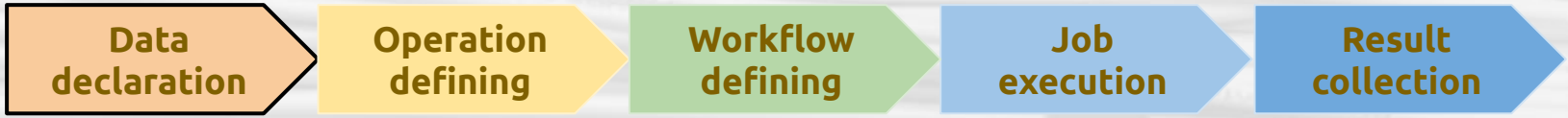
inner dimension

Parameters for pre-processing, metadata, and definition etc.

retrieve = TRUE ⟶ Load data in workstation and occupy memory

retrieve = FALSE ⟶ Create a pointer to data repository and obtain metadata only

1. Data declaration



Start() parameters

Help users organize data structure with simply a few lines.

[define dimension]
pattern_dims
metadata_dims
path_glob_permissive
return_vars
synonims
*_depends
*_across
*_var

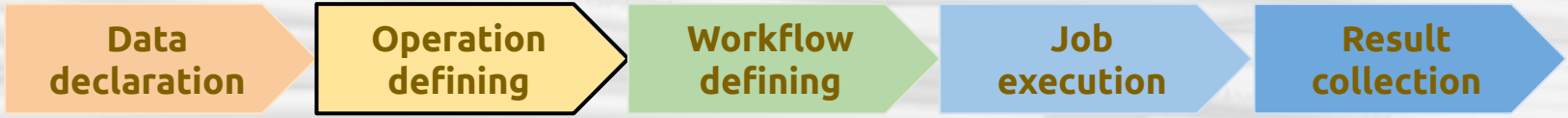
[reshape]
merge_across_dims
merge_across_dims_narm
split_multiselected_dims

[interpolate]
transform
transform_params
transform_vars
transform_extra_cells
apply_indices_after_transform

[interface function]
file_opener
file_var_reader
file_dim_reader
file_data_reader
file_closer
selector_checker

[operation]
num_procs
silent
debug

2. Operation defining



- Define the operation in the **R function format**.
- The operation is only for **essential dimensions** but not the whole data, which is the concept of multiApply.
- The output size should be small enough to fit in the workstation.

E.g., Start() call detected the data size:

Detected dimension size:

dat	var	sdate	member	time	latitude	longitude
1	1	32	25	7	640	1296

Total size of involved data:

* 1 x 1 x 32 x 25 x 7 x 640 x 1296 x 8 bytes = **34.6 Gb**

Too large to fit in workstation. The operation has to reduce the size.

2. Operation defining

Data declaration → **Operation defining** → Workflow defining → Job execution → Result collection

Detected dimension size:

dat	var	sdate	member	time	latitude	longitude
1	1	32	25	7	640	1296

Total size of involved data:

* 1 x 1 x 32 x 25 x 7 x 640 x 1296 x 8 bytes = **34.6 Gb**

Do ensemble mean and calculate temporal trend

Expected output dimension size:

dat	var	sdate	member	time	latitude	longitude
1	1	1	1	7	640	1296

Total size of expected output:

* 1 x 1 x 1 x 1 x 7 x 640 x 1296 x 8 bytes = **44.3 Mb**

```
fun <- function(x) {
  # x: [sdate, member]
  # ensemble mean
  x <- apply(x, 1, mean)
  # trend
  x <- s2dv:::.Trend(x)$trend[2]
  return(x)
}
```

Small enough to fit in workstation and do the following operation (e.g., plotting).

3. Workflow defining

Data declaration → Operation defining → **Workflow defining** → Job execution → Result collection

- Identify the dimensions to be operated on and the expected output dimensions.
- Join the data (i.e., the startR_cube object from `Start()`) and the user-defined function together.

```
step <- Step(fun = fun,
             target_dims = c('sdate', 'member'),
             output_dims = NULL)

wf <- AddStep(data, step, ...)
```

workflow is built!

Which dimensions the operation performs on?
Which dimensions of output are expected?

(Check the function in the previous step)

```
fun <- function(x) {
  # x: [sdate, member]
  # ensemble mean
  x <- apply(x, 1, mean)
  # trend
  x <- s2dv:::.Trend(x)$trend[2]
  return(x)
}
```

4. Job execution

Data declaration | Operation defining | Workflow defining | **Job execution** | Result collection

- Execute the workflow either locally or on HPCs
- Decide the chunking. Ensure that each chunk size fits in the memory module of HPC

```
res <- Compute(wf,
               chunks = list(latitude = 2, longitude = 2),
               threads_load = 2, threads_compute = 4,
               cluster = list(
                 queue_host = 'nord3',
                 queue_type = 'lsf',
                 temp_dir = '/gpfs/scratch/bsc32/
                 cores_per_job = 2,
                 job_wallclock = '05:00',
                 max_jobs = 4,
                 extra_queue_params = list('#BSUB -q bsc_es'),
                 bidirectional = FALSE,
                 polling_period = 10,
               ...),
               ecflow_suite_dir = '/home/Earth/user_id/startR_local/',
               wait = TRUE)
```

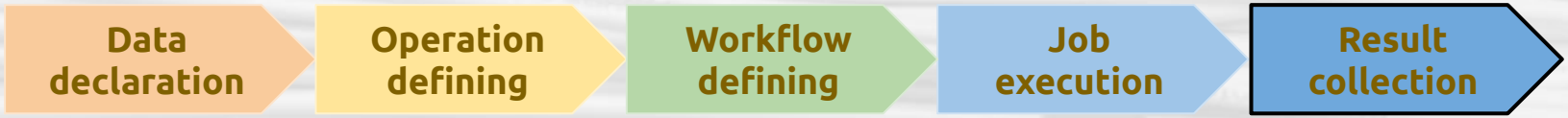
Cluster configuration

34.6 Gb is divided into 4 chunks

Detected dimension size:

dat	var	sdate	member	time	latitude	longitude
1	1	32	25	7	640	1296

5. Result collection



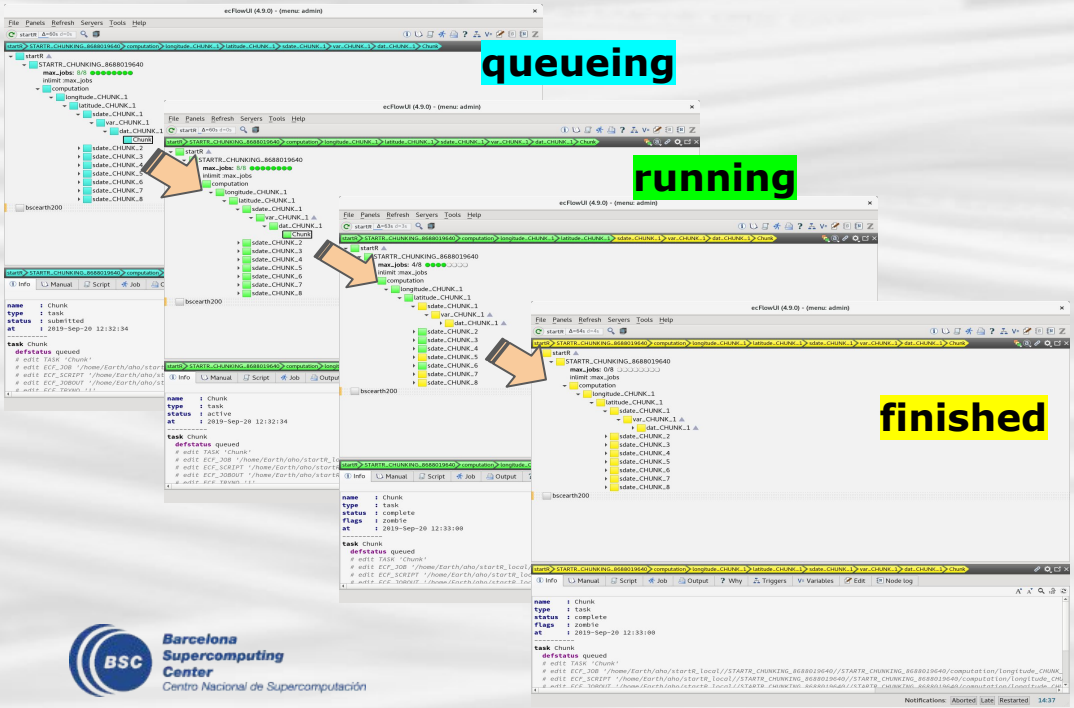
- The results of each chunk combine and return automatically if we wait for the execution finished.
- For the long computation we don't wait, `Collect()` is used to combine and return the results back to the workstation.

```
res <- Compute(wf,
               …,
               wait = FALSE)

saveRDS(res, file = './res_collect.Rds') ⟶ Store the descriptor of the execution
**Now you can close the R console and come back later**

collect_info <- readRDS('./res_collect.Rds')
result <- Collect(collect_info, wait = TRUE)
```

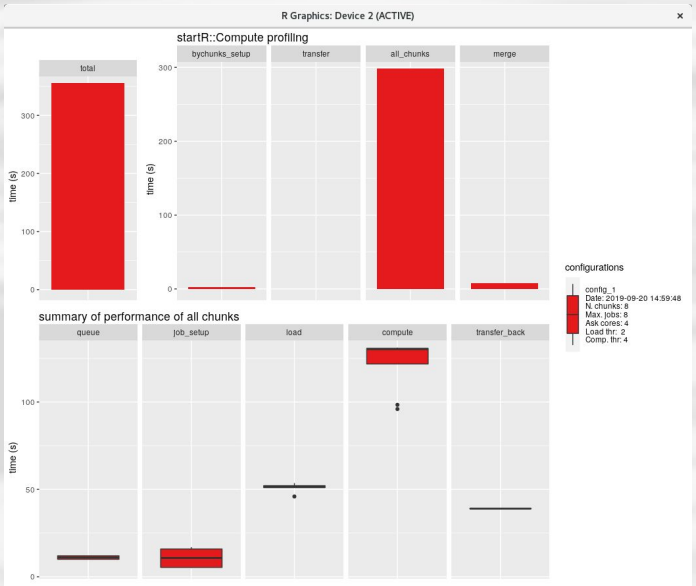
Monitor and profile the execution

Data declaration → Operation defining → Workflow defining → Job execution → Result collection

Monitoring the execution through **ecFlow UI**

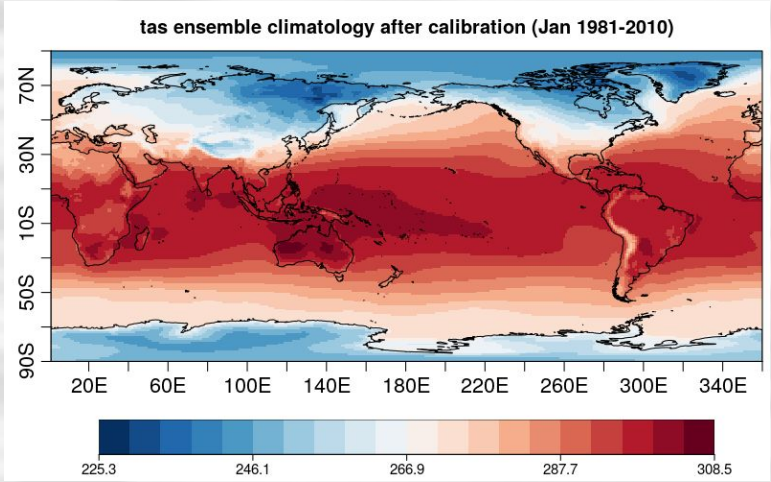


Profiling the execution



Use case: Calibration and Climatology

- Data:
  - tas/monthly_mean/January 1981 - 2010
  - experiment: ECMWF/system5_m1
  - observation: ECMWF/Era-interim
- Pre-processing
  - Reshape the dimensions
  - Regrid to 1° resolution
  - Reorder latitudes
- Operations:
  - Bias adjustment. Use **CSTools:::.cal** (the interior function of **CST_Calibration)**
  - Ensemble mean
  - Monthly climatology



Use case: Calibration and Climatology

1. Data declaration

Original data structure of netCDF file:

exp

* dat: 1
* var: 1
* sdate: 360
* ensemble: 25
* time: 1
* latitude: 640
* longitude: 1296

obs

* dat: 1
* var: 1
* sdate: 360
* time: 1
* lat: 256
* lon: 512

Data structure after pre-processing by `Start()`:

exp

* dat: 1
* var: 1
* smonth: 12
* syear: 30
* ensemble: 25
* time: 1
* latitude: 181
* longitude: 360

obs

* dat: 1
* var: 1
* smonth: 12
* syear: 30
* time: 1
* latitude: 181
* longitude: 360

Consistent!

Use case: Calibration and Climatology

Experimental data

1. Data declaration

```
# Declaration of data sources
exp <- Start(dat = '/esarchive/exp/ecmwf/system4_m1/monthly_mean/$var$_f6h/$var$_$sdate$.nc',
             var = 'tas',
             sdate = sdates,
             split_multiselected_dims = TRUE,
             ensemble = 'all',
             time = indices(1),
             latitude = values(list(lats.min, lats.max)),
             latitude_reorder = Sort(),
             longitude = values(list(lons.min, lons.max)),
             longitude_reorder = CircularSort(0, 360),
             transform = CDORemapper,
             transform_extra_cells = 2,
             transform_params = list(grid = 'r360x181',
                                     method = 'conservative',
                                     crop = c(lons.min, lons.max, lats.min, lats.max)),
             transform_vars = c('latitude', 'longitude'),
             return_vars = list(latitude = 'dat',
                                longitude = 'dat',
                                time = c('sdate')),
             retrieve = FALSE)
```

reshape

reorder

regrid

metadata

Use case: Calibration and Climatology

Observational data *Make the array structure consistent with experimental one.*

1. Data declaration

```
obs <- Start(dat = '/esarchive/recon/ecmwf/erainterim/monthly_mean/$var$_f6h/$var$_$sdate$.nc',
             var = 'tas',
             sdate = sdates_obs,
             split_multiselected_dims = TRUE,
             time = indices(1),
             latitude = values(list(lats.min, lats.max)),
             latitude_reorder = Sort(),
             longitude = values(list(lons.min, lons.max)),
             longitude_reorder = CircularSort(0, 360),
             transform = CDORemapper,
             transform_extra_cells = 2,
             transform_params = list(grid = 'r360x181',
                                     method = 'conservative',
                                     crop = c(lons.min, lons.max, lats.min, lats.max)),
             transform_vars = c('latitude', 'longitude'),
             synonims = list(latitude = c('latitude', 'lat'),
                             longitude = c('longitude', 'lon')),
             return_vars = list(latitude = 'dat',
                                longitude = 'dat',
                                time = c('sdate')),
             retrieve = FALSE)
```

reshape

reorder

regrid

rename

metadata

Use case: Calibration and Climatology

2. Operation defining

```
# Define the workflow
## Function
wrap_cal <- function(obs, exp) {
    calibrated <- CSTools:::.cal(exp = exp, obs = obs,
                                 cal.method = "bias", eval.method = "leave-one-out",
                                 multi.model = FALSE, na.fill = TRUE, na.rm = TRUE,
                                 apply_to = NULL, alpha = NULL)  # [ensemble, syear]
    #ensemble mean
    ens_mean <- apply(calibrated, 2, mean, na.rm = TRUE)
    #climatology
    clim <- mean(ens_mean, na.rm = TRUE)
    return(clim)
}
```

R function format

calibration

ensemble mean

climatology

3. Workflow defining

```
## Workflow
step <- Step(wrap_cal,
             target_dims = list(obs = c('syear'), exp = c('ensemble', 'syear')),
             output_dims = NULL)

wf <- AddStep(list(obs = obs, exp = exp), step)
```

Total size of involved data:
exp: 1 x 1 x 360 x 25 x 1 x 640 x 1296 x 8 bytes = **55.6 Gb**
obs: 1 x 1 x 360 x 1 x 256 x 512 x 8 bytes = **360 Mb**

Estimated size of output data:
1 x 1 x 12 ~~x 30 x 25~~ x 1 x 181 x 360 x 8 bytes = **6.25 Mb**

Use case: Calibration and Climatology

Compute on Nord3 cluster

4. Job execution

```
#-------------------user-defined---------------------
  queue_host <- 'nord1'
  temp_dir <- '/gpfs/scratch/bsc32/bsc32734/startR_hpc/'
  ecflow_suite_dir <- '/home/Earth/aho/startR_local/'
#------------------------------------------------------

  res <- Compute(wf,
                 chunks = list(smonth =12),
                 threads_load = 2,
                 threads_compute = 4,
                 cluster = list(queue_host = queue_host,
                   queue_type = 'lsf',
                   temp_dir = temp_dir,
                   cores_per_job = 2,
                   job_wallclock = '05:00',
                   max_jobs = 4,
                   extra_queue_params = list('#BSUB -q bsc_es'),
                   bidirectional = FALSE,
                   polling_period = 10
                 ),
                 ecflow_suite_dir = ecflow_suite_dir,
                 wait = TRUE
                 )  #$output1
```

Chunking

Nord3 configuration

Workflow comparison

startR workflow

```
repos <- paste0('/esarchive/exp/ecmwf/system4_m1/monthly_mean/',
                '$var$_f6h/$var$_$sdate$.nc')
sdates <- sapply(2012:2016, function(x) paste0(x, sprintf('%02d', 1:12), '01'))

# exp data
exp <- Start(dat = repos,
             var = 'tas',
             sdate = sdates,
             time = indices(1),
             ensemble = 'all',
             latitude = 'all',
             longitude = 'all',
             synonims = list(longitude = c('lon', 'longitude'),
                             latitude = c('lat', 'latitude')),
             retrieve = F)

# obs data
repos_obs <- paste0('/esarchive/recon/ecmwf/erainterim/monthly_mean/',
                    '$var$_f6h/$var$_$sdate$.nc')
sdates_obs <- (sapply(2012:2016, function(x) paste0(x, sprintf('%02d', 1:12))))

obs <- Start(dat = repos_obs,
             var = 'tas',
             sdate = sdates_obs,
             time = indices(1),
             latitude = 'all',
             longitude = 'all',
             synonims = list(longitude = c('lon', 'longitude'),
                             latitude = c('lat', 'latitude')),
             retrieve = F)
```

```
#Calculate the ensemble-adjusted Continuous Ranked Probability Score (CRPS)
func <- function(x, y) {
  crps <- mean(SpecsVerification::EnsCrps(x, y, R.new = Inf))
  return(crps)
}
step <- Step(func, target_dims = list(c('sdate', 'ensemble'), c('sdate')),
             output_dims = NULL)
wf <- AddStep(list(exp, obs), step)
```

```
# Power 9
res <- Compute(wf,
               chunks = list(latitude = 2,
                             longitude = 2),
               threads_load = 2,
               threads_compute = 4,
               cluster = list(queue_host = 'p1',
                              queue_type = 'slurm',
                              temp_dir = '/gpfs/scratch/bsc32/bsc32734/startR_hpc/',
                              r_module = 'R/3.5.0-foss-2018b',
                              job_wallclock = '00:30:00',
                              cores_per_job = 4,
                              max_jobs = 4,
                              bidirectional = FALSE,
                              polling_period = 50),
               ecflow_suite_dir = '/home/Earth/aho/startR_local/',
               wait = TRUE
               )  #$output1
```

Declare large data easily

Well-organized and concise

chunking automatically & parallel computing

without startR workflow

chunking manually

pre-processing manually

Serial computing locally

Mixed steps

```
path_exp <- list(name = 'system4_m1',
                 path = file.path('/esarchive/exp/ecmwf',
                                  '$EXP_NAME$/monthly_mean',
                                  '$VAR_NAME$_f6h/$VAR_NAME$_$START_DATE$.nc'))

path_obs <- list(name = 'erainterim',
                 path = file.path('/esarchive/recon/ecmwf',
                                  '$OBS_NAME$/monthly_mean',
                                  '$VAR_NAME$_f6h/$VAR_NAME$_$YEAR$$MONTH$.nc'))

res <- array(NA, dim = c(year = 5, month = 2, dataset = 1,
                         ftime = 1, lat = 256, lon = 512, member = 1))

t_begin_cal <- Sys.time()
```

```
# Do chunking manually
for (yr in 2012:2016) {
  for (mth in 0:1) {
    startDates <- sapply(1:5, function(x) paste0(x,
                                                 sprintf('%02d', (1 + 6 * mth):(6 + 6 * mth)), '01'))

    data <- Load(var = 'tas',
                 exp = list(path_exp),
                 obs = list(path_obs),
                 sdates = startDates,
                 storefreq = "monthly",
                 leadtimemax = 1,
                 output = 'lonlat')

    mod <- data$mod
    obs <- data$obs
```

```
    #Calculate the ensemble-adjusted Continuous Ranked Probability Score (CRPS)
    CRPS <- multiApply::Apply(data = list(mod, obs),
                              target_dims = list(c('sdate', 'member'),
                                                 c('sdate')),
                              fun = SpecsVerification::EnsCrps)$output1
    names(dim(CRPS))[1] <- c('sdate')
    CRPS <- multiApply::Apply(data = CRPS,
                              target_dims = c('sdate'),
                              fun = mean)$output1

    res[yr-2011, mth+1, , , , , ] <- CRPS
  } # mth for loop
} # yr for loop

CRPS <- multiApply::Apply(data = res,
                          target_dims = c('month'),
                          fun = mean)$output1
CRPS <- multiApply::Apply(data = CRPS,
                          target_dims = c('year'),
                          fun = mean)$output1

t_end_cal <- Sys.time()
t_cal <- as.numeric(difftime(t_end_cal, t_begin_cal, units = 'mins'))
#[1] 8.041727 mins
```

Summary and future work

startR's advantages:

- Proficient in big and complex multi-dimensional data retrieval and processing
- Highly adaptable to data structure and users' needs
- Clear and concise workflow, easy to be reused and adapted to other analyses
- Automatically chunking and dispatching jobs in parallel on HPCs
- Compatible with other R tools developed in the department, forming a strong toolset for climate research
- With the plug-in of interface functions, startR can be exploited in different scientific domains where large multi-dimensional data is involved

startR's disadvantages:

We provide resources and support!

- Long learning curve
- Not-so-intuitive coding style (functional paradigm)

Summary and future work

Applications:

- S2S4E project *link*
- ESMValTool, C3S MAGIC project *link*
- Individual research

Future work:

- Increase the flexibility of the retrieval of different datasets
- Multiple steps in the workflow
- Retrieve data from the cloud

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Relative speech

Tomorrow 10:20h Climate Forecast Analysis Tools Framework by Núria Pérez-Zanón

Resources on GitLab

- **Documentation**
  README.md, pracitcal_guide.md, faq.md… etc.
- **Example scripts:** usecase.md

Resources on CRAN

- Find the manual and installation here

Contact

- **Núria Pérez-Zanón** (nuria.perez@bsc.es)
- **An-Chi Ho** (an.ho@bsc.es)

Thank you and let's startR!