Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

07/10/2021

R tools users meeting

An-Chi Ho and Núria Pérez-Zanón

contributor:

Agenda

1. Icebreaker: CDO
2. News
    - startR
        - Regrid + chunk
        - Regridding in Start() works with selector 'all' and indices()
    - s2dv
        - PlotEquiMap(): country boundary and shapefile
        - CDORemap(): bugfixes for irregular regridding
    - CSTools
3. Discussion:
    - Lat-level interpolation
    - Subseasonal use case
4. Q&A

startR

(recent internal release: v2.1.0-5)

New resources

To deal with the files that do not have consistent data structure:

- Use an array of actual values as the inner dim selector to specify different value for each file.
  - FAQ: https://earth.bsc.es/gitlab/es/startR/-/blob/master/inst/doc/faq.md#22-define-the-selector-when-the-indices-in-the-files-are-not-aligned
  - Use case: https://earth.bsc.es/gitlab/es/startR/-/blob/master/inst/doc/usecase/ex1_13_implicit_dependency.R
- Use case: Use actual values to define two file dimensions that has dependency. https://earth.bsc.es/gitlab/es/startR/-/blob/master/inst/doc/usecase/ex1_14_file_dependency.R

Check regridding data:
https://earth.bsc.es/gitlab/es/startR/-/blob/master/inst/doc/data_check.md#5-regridding

transform with ‘all’ and indices()

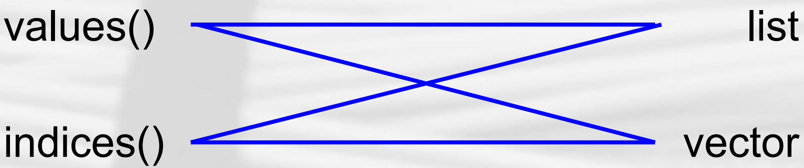
```
data <- Start(dat = path,
              var = 'tos',
              sdate = ‘20000101’,
              time = 'all',
              lat =
              lon =
              lat_reorder = Sort(),
              lon_reorder = CircularSort(0, 360),
              transform = CDORemapper,
              transform_extra_cells = 2,
              transform_params = list(grid = 'r100x50',
                                      method = 'con',
                                      crop =  c(0, 360, -90, 90)),
              transform_vars = c('lat','lon'),
              synonims = list(lon = c('lon', 'longitude'), lat = c('lat', 'latitude')),
              return_vars = list(lat = 'dat', lon = 'dat', time = 'sdate'))
```

‘all’	indices(1:640)	values(list(-90, 90))
‘all’	indices(1:1296)	values(list(0, 359.9))

Reminder: “_reorder” is recommended using always.

transform with ‘all’ and indices()

- FAQ: The best practice of using vector and list for selectors https://earth.bsc.es/gitlab/es/startR/-/blob/master/inst/doc/faq.md#23-the-best-practice-of-using-vector-and-list-for-selectors



values()

list

indices()

vector

→ 4 possibilities

- Recommendation:
  - latitude and longitude - use values(list(a, b))
  - file dimensions - use vector of values,
    e.g., sdate = c(‘200001’, ‘200101’, ‘200201’)
  - inner dimensions - use vector of indices or values,
    e.g., time = indices(1:12);
    time = as.POSIXct(times * 86400, tz = 'UTC', origin = '1970-01-01')

CDO usage in CDORemap

- cdo -s (sellonlatbox) remap method, grid input_file output_file

  e.g.,

  cdo -s remapycon,r100x50 /tmp/input.nc /tmp/output.nc

  cdo -s sellonlatbox,0,359.9,-90,90 -remapycon,r100x50 /tmp/input.nc /tmp/output.nc\

- Before this line, we need to prepare the input .nc file…
  - Take a data array, organize it, and save it as a temporary netCDF file.
  - Consider regular or irregular grid.
    - Subset the array if needed (when dimension number > 4)
    - Reorder the dimensions if needed
    - Check dimension attributes (limited/unlimited)
    - metadata

CDO usage in CDORemap

- After regridding, open the output netCDF file and organize the array again
  - reorder and rename the dimensions
  - restore metadata of the original data array

‘transform’ in Start()

```
exp <- Start(...,
        transform = CDORemapper,
        transform_params = list(grid = 'r100x50',
                                method = 'con',
                                crop = F),
        transform_vars = c('latitude', 'longitude'),
        transform_extra_cells = 8,
        …)
```

‘con’, ‘bil’, ‘bic’, ‘dis’

FALSE, TRUE,
c(lon.min, lon.max,.lat.min, lat.max)

Extra grids extended from the borders for interpolation

These parameters can affect the regridding result.

Regrid + chunk over latitude and longitude

```
exp <- Start(dat = path,
          sdate = '20000101',
          ensemble = indices(1),
          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_params = list(grid = 'r100x50',
                                  method = 'con',
                                  crop = F),
          transform_vars = c('latitude', 'longitude'),
          transform_extra_cells = 8,
          synonims = list(latitude = c('lat', 'latitude'), longitude
= c('longitude', 'lon')),
          return_vars = list(latitude = NULL, longitude = NULL,
time = 'sdate'),
          retrieve= F)
```

regrid

```
func <- function(exp) {
  return(exp)
}
step <- Step(func,
             target_dims = 'sdate',
             output_dims = 'sdate')
wf <- AddStep(exp, step)

res1 <- Compute(wf,
                chunks = list(latitude = 2,
                              longitude = 2))
```

chunk

"Regrid + chunk" is a new feature of startR. Before, you can only chunk the dimensions that are neither the target dimensions nor lat/lon if regridding is applied.
FAQ:
https://earth.bsc.es/gitlab/es/startR/-/blob/master/inst/doc/faq.md#24-do-both-interpolation-and-chunking-on-spatial-dimensions

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

How many extra cells are needed for transformation?

The parameter “transform_extra_cells” can be used to extended the grid points for transformation to avoid the inaccuracy at the border (See FAQ How-to-5 for more details.)

The default value of “transform_extra_cells” is 2, which is consistent with s2dv::Load. HOWEVER, it may not be the best number in some cases.

How many extra cells are needed for transformation?

[Example case]

- Selected region: longitude is [10:20]; latitude is [20:40]
- Original resolution: 1296x640
- Target resolution: 100x50
- transform method: ‘con’
- transform crop: c(10, 20, 20, 40)

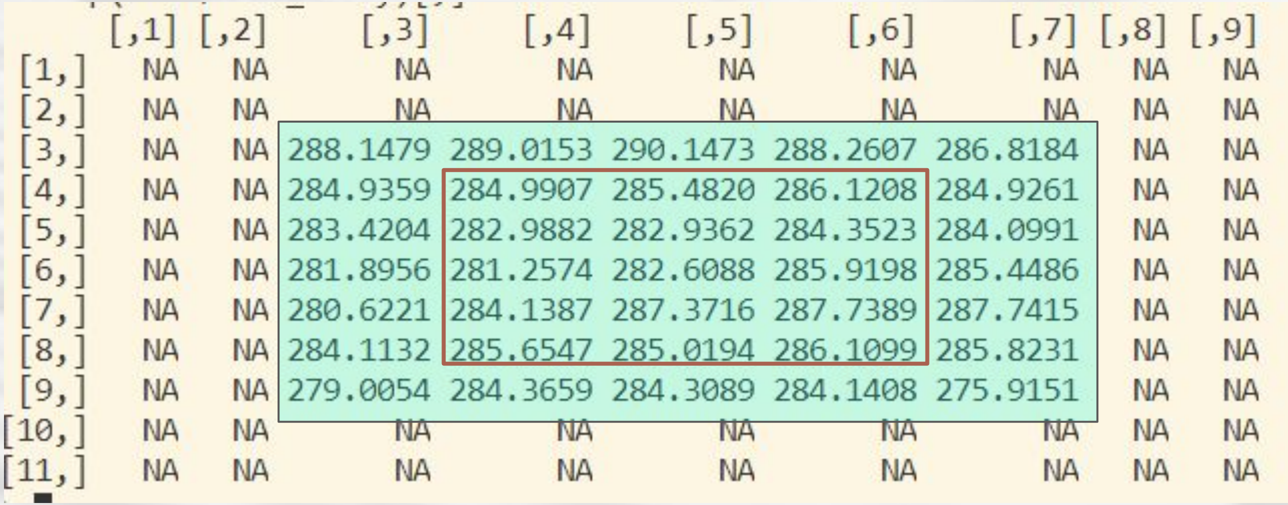
In this case, “transform_extra_cells” needs to be at least 4 to get the most accurate values* at longitude left border. Whereas, 2 extra cells are enough for longitude right border as well as for latitude.

**”The most accurate value” means the regridding of global domain.*

How many extra cells are needed for transformation?

[Example case]

Use CDORemap() to testify the results. Put 4 extra cells to transformation, and the matrix is like:



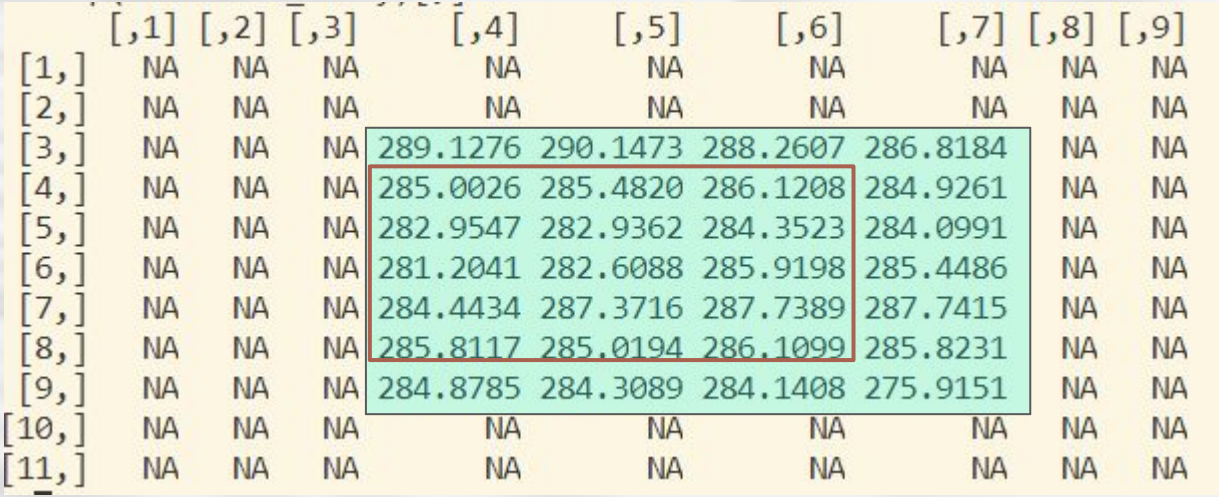
```
      [,1] [,2]     [,3]     [,4]     [,5]     [,6]     [,7] [,8] [,9]
 [1,]   NA   NA       NA       NA       NA       NA       NA   NA   NA
 [2,]   NA   NA       NA       NA       NA       NA       NA   NA   NA
 [3,]   NA   NA 288.1479 289.0153 290.1473 288.2607 286.8184   NA   NA
 [4,]   NA   NA 284.9359 284.9907 285.4820 286.1208 284.9261   NA   NA
 [5,]   NA   NA 283.4204 282.9882 282.9362 284.3523 284.0991   NA   NA
 [6,]   NA   NA 281.8956 281.2574 282.6088 285.9198 285.4486   NA   NA
 [7,]   NA   NA 280.6221 284.1387 287.3716 287.7389 287.7415   NA   NA
 [8,]   NA   NA 284.1132 285.6547 285.0194 286.1099 285.8231   NA   NA
 [9,]   NA   NA 279.0054 284.3659 284.3089 284.1408 275.9151   NA   NA
[10,]   NA   NA       NA       NA       NA       NA       NA   NA   NA
[11,]   NA   NA       NA       NA       NA       NA       NA   NA   NA
```

The result is correct because neither of these points are at the border.

How many extra cells are needed for transformation?

[Example case]

Use CDORemap() to testify the results. Put 2 extra cells to transformation, and the matrix is like:

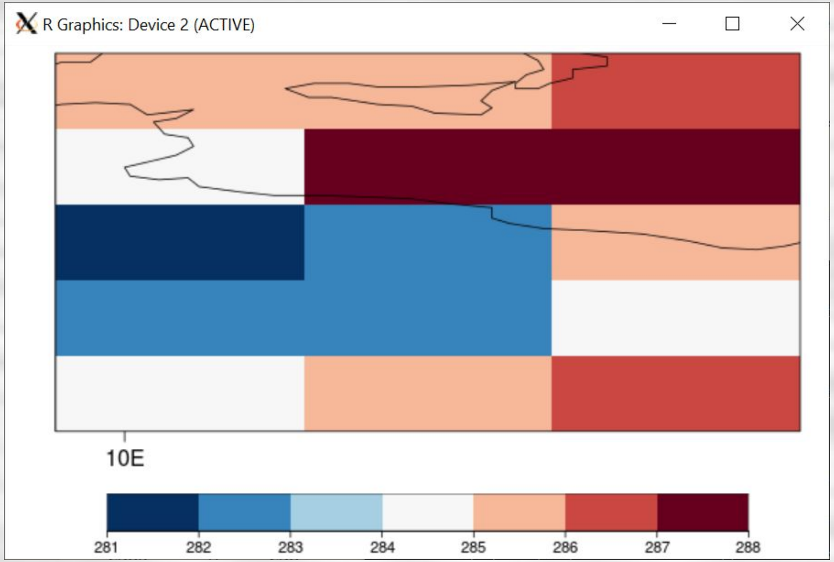


```
      [,1] [,2] [,3]     [,4]     [,5]     [,6]     [,7] [,8] [,9]
 [1,]   NA   NA   NA       NA       NA       NA       NA   NA   NA
 [2,]   NA   NA   NA       NA       NA       NA       NA   NA   NA
 [3,]   NA   NA   NA 289.1276 290.1473 288.2607 286.8184   NA   NA
 [4,]   NA   NA   NA 285.0026 285.4820 286.1208 284.9261   NA   NA
 [5,]   NA   NA   NA 282.9547 282.9362 284.3523 284.0991   NA   NA
 [6,]   NA   NA   NA 281.2041 282.6088 285.9198 285.4486   NA   NA
 [7,]   NA   NA   NA 284.4434 287.3716 287.7389 287.7415   NA   NA
 [8,]   NA   NA   NA 285.8117 285.0194 286.1099 285.8231   NA   NA
 [9,]   NA   NA   NA 284.8785 284.3089 284.1408 275.9151   NA   NA
[10,]   NA   NA   NA       NA       NA       NA       NA   NA   NA
[11,]   NA   NA   NA       NA       NA       NA       NA   NA   NA
```

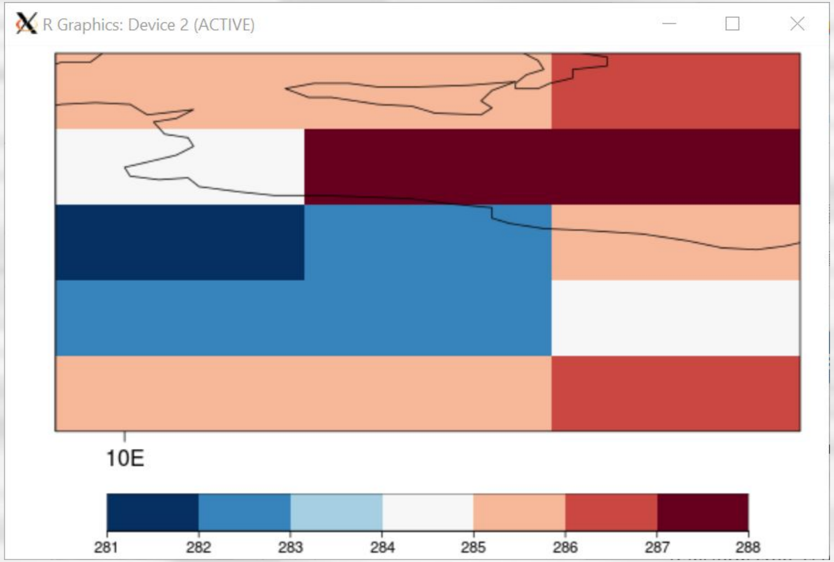
The result is NOT correct because the longitude are at the left border.

How many extra cells are needed for transformation?

[Example case]



extra cells = 4



extra cells = 2

The difference is not big but may still have certain impacts.

How many extra cells are needed for transformation?

- Possible factors:
  - Original grid resolution
  - Target grid resolution
  - method (See the algorithm of different method in the CDO manual)
  - crop option (region, TRUE, FALSE)
  - Selected region (i.e., how the selected borders locate at original and target grids)
  - ?
- For global domain, 2 extra cells are enough (from my experience). BUT if chunking along lat/lon, it may not be enough.
- The safe way is to select the domain larger than the desired one. But if chunking is applied, the impact at borders is inevitable.
- crop = c(lon.min, lon.max, lat.min, lat.max) is recommended.

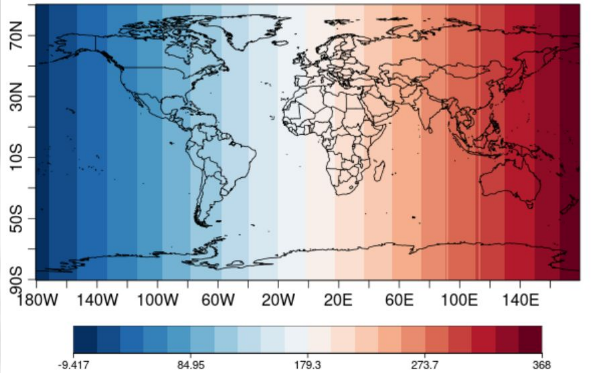
s2dv

PlotEquiMap: Plot country boundary & add shape file

New arguments:

- **country.borders**: A logical value indicating if the country borders should be plotted (TRUE) or not (FALSE). It only works when 'filled.continents' is FALSE. The default value is FALSE.
- **shape_file**: A character string of the path to the shape file. The format should be ".rds". If it is specified, 'country.borders' won't be used. The default value is NULL.

The development is in branch develop-PlotEquiMap_country_border now.



CDORemap: bugfixes for irregular regridding

Some remaining bugs (swapped lat and lon) of regridding the irregular grid by CDORemap() has been fixed. If you’re interested in it, the fixed function is under branch develop-CDORemap_irregular now.

CSTools 4.0.1

CSTools 4.0.1

- CSTools 4.0.1 is on its way to CRAN https://CRAN.R-project.org/package=CSTools

- NEWS available https://cran.r-project.org/web/packages/CSTools/news/news.html

- CSTools 4.0.1 is being shared on Zenodo to get a DOI number: 10.5281/zenodo.5549474 to be part of the manuscript under preparation as requested by the GMD journal (Geoscientific Model Development)

- It is installed in workstations R/3.6.1 and Nord3 R/3.6.2

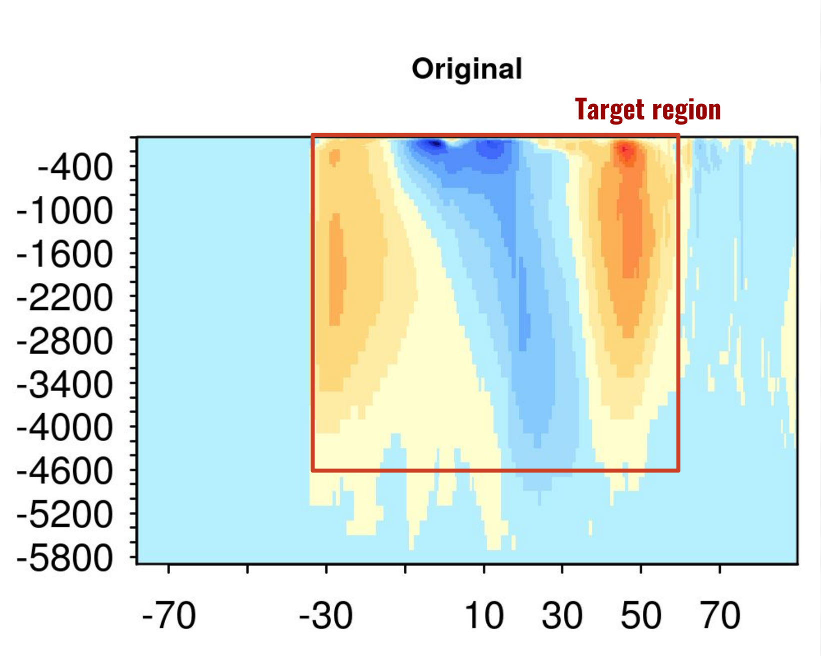
Lat-level interpolation
?

Lat-level interpolation

The objective is to interpolate this 2D data

Original coordinates not equally spaced
x: latitudinal coordinates [292]
[range: -78.0132294, 89.5945358]
y: depth [75]
[range: 0, 5800] 0, 1.2, …, 102.42, 113.88,...

Target coordinates
x: latitudinal coordinates [292] regularly spaced
[range: -30, 60]
y: depth [75] irregularly spaced
[range: 0, 4500] 5,10,...200,300,..,4250, 4500

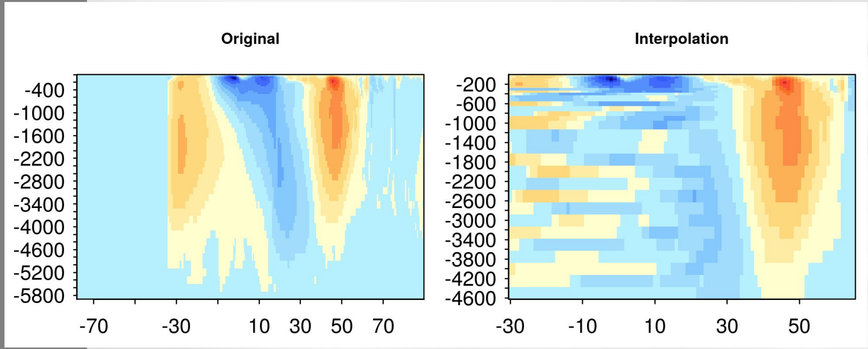


Lat-level interpolation

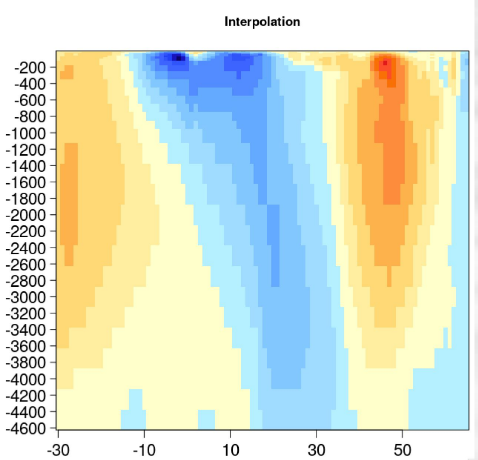
```
library(akima)
...
inter_lin = interp(lats, levels, t(diff_ecearth_yr_ens_mvolc),
              xo = lat1,
              yo = depth1,
              linear = TRUE, extrap = FALSE)

PlotSection(diff_ecearth_yr_ens_mvolc, rlat_ecearth,
lev_ecearth, toptitle = 'Original', cols=jBrewColorsano,
brks=brks_def, drawleg=F, sizetit = 0.5)
```

Linear = TRUE



Linear = FALSE



Linear = TRUE only valid for regular grids

https://CRAN.R-project.org/package=akima

Hiroshi Akima, "A Method of Bivariate Interpolation and Smooth Surface Fitting for Irregularly Distributed Data Points", ACM Transactions on Mathematical Software, Vol. 4, No. 2, June 1978, pp. 148-159. Copyright 1978, Association for Computing Machinery, Inc., reprinted by permission.

BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Lat-level interpolation

- Is this method correct?
- Does any one know another method/package to do this?
- How many people need this kind of code?
- Where do you think is the best place to share this code? (use case/repo/wiki??)

Subseasonal verification use case using startR

Subseasonal verification

During the R users meeting on the 7th of May 2021, Andrea showed us a subseasonal forecast analysis: https://earth.bsc.es/wiki/lib/exe/fetch.php?media=tools:r_user_meeting_20210507.pdf

Now we are working to create a use case for the startR documentation: https://earth.bsc.es/gitlab/es/startR/-/blob/develop-verification/inst/doc/usecase.md

In which it is explained:

- how the hindcast forecast of ECMWF-S2S files are stored
- how to load the hindcast and the reference dataset for the corresponding dates
- how to calculate verification skill scores
- how to submitted to a HPC cluster with Compute()
- how to interprete the results

We plan to create new use cases for forecast verification, because

- it can help to replicate results or procedures
- it can help newcomers to learn startR and forecast verification
- it can help to test the operational results or debugging its code
- As tools developers, it can help us to understand how do you use other tools (e.g.: s2dv)

- How useful do you think it can be?
- Do you have any other verfication script?
- What about other forecast horizons?
- What about verification of Indicators?

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Q & A

Next meeting: 4th Nov. 2021 (11 am)