startR: A tool for large multi-dimensional data processing

Núria Pérez-Zanón[*1], An-Chi Ho[*2], Nicolau Manubens[*3], Francesco Benincasa[*4],
Pierre-Antoine Bretonnière[*5]

[*]*Barcelona Supercomputing Center, Barcelona, Spain*
[1]nuria.perez@bsc.es, [2]an.ho@bsc.es, [3]nicolau.manubens@bsc.es, [4]francesco.benincasa@bsc.es,
[5]pierre-antoine.bretonniere@bsc.es

***Keywords*— big data, data processing, earth sciences, high-performance computing, high-dimensional data**

EXTENDED ABSTRACT

Nowadays, the huge amount of data produced in various scientific domains has made data analysis challenging. In the climate science domain, with the constant increase of resolution in all possible dimensions of model output and the growing need for using computationally demanding analytical methodologies (e.g. bootstrapping), basic operations like extracting data from storage and performing statistical analysis on them must fulfill scientific and operational needs taking into account the growing volume and variety of data. A tool that facilitates data processing and leverages computational resources can largely save researchers' time and effort.

In this work, we introduce startR, an R package that allows to retrieve, arrange, and process large multi-dimensional datasets automatically with a concise workflow, smoothing the data-processing difficulty mentioned above.

A. Introduction

startR provides a framework under which users can apply a self-defined procedure to a collection of multi-dimensional datasets from the repository as large as desired and take advantage of the computational resources in high-performance computing systems (HPC). It has been designed to require as few technical parameters as possible from the users, but users can tailor a number of configurations to adjust the execution according to their needs.

Under this framework, two approaches to conduct analysis are available: retrieving the data into RAM and performing the analysis, or creating a workflow -the data is not retrieved only declared- in which the last step is to run the user-defined analysis. The first one allows the user to explore the analysis at any stage whereas, in the second one, larger data can be involved in the analysis for the same available RAM. When the workflow is preferred, all the information needed for the data analysis can be encapsulated in a succinct and reusable script which is composed of four main functions: `Start`, `Step`, `AddStep`, and `Compute` (Fig. 1). The required information is all described and defined by these functions, including the involved data file distribution, the dimensions of the desired data cube, the workflow of operations to be applied, the job execution parameters and HPC configuration if needed.

The object type startR works with is array with named dimensions, a basic and widely used data structure. startR obtains data from the repository and returns an array object. The array can have an arbitrary number of dimensions and the order is not restricted as well since startR works with the dimension names rather than the index. This feature provides flexibility for data structure and makes the analysis less error-prone. When the execution finishes, startR also returns an array with named dimensions that can be processed further or stored in files for later use. Besides the data itself, metadata is also well-preserved and expanded with the operation information, ensuring the reproducibility of the analysis.

The startR framework implements the MapReduce paradigm for chunking and processing big data sets. The execution can run either locally or remotely on HPCs with multi-node and multi-core parallelism where possible. The EC-Flow workflow manager is used to dispatch tasks onto the HPCs, providing fault-tolerance and progress control through its graphic user interface (GUI).

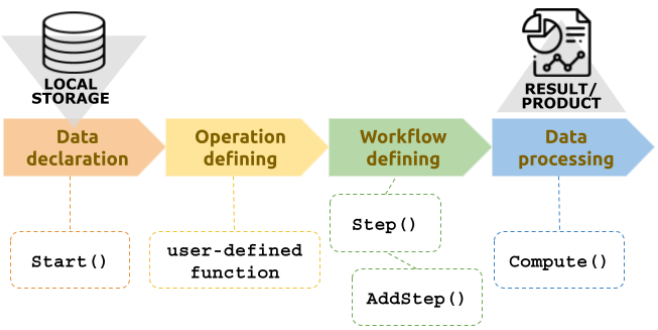

Fig. 1 The startR workflow and the corresponding functions

B. Data declaration

The first step of data analysis is extracting data from storage, which is achieved by function `Start`. startR can access the repositories and the data files do not need to comply with any specific convention for their distribution pattern, file names, or the number and order of the dimensions of the variables. Though netCDF is the only data format supported in the current release, adaptors for other file formats can be plugged in startR if needed.

Start() declares the requested data as an array object. It provides two options for data retrieval: loading the entire data set in the machine or creating a pointer to the data location in the repository. With the first option, the Start call returns a multi-dimensional array that occupies the memory space of the workstation, so the involved data size is limited. On the other hand, the Start call only retrieves the dimensions and metadata of the required data and the storage location with the latter option, so the involved data size is unlimited in theory. Loading data is the conventional way for the following analysis, whereas generating a pointer can make full use of the functionality of startR.

There are several advanced parameters in Start() to handle heterogeneities across files involved in one Start() call, such as transformation, reordering, reshaping, and renaming (Table 1). In addition, ancillary data and metadata can be well-preserved through parameters considering the complex data file distribution. Start() also accepts user-defined functions for some of these purposes.

TABLE I
START() PARAMETERS AND THEIR FUNCTIONALITY

Functionality	Parameters
Transformation: Interpolate data to the desired grid type.	transform transform_params transform_vars transform_extra_cells apply_indices_after_transform
Reshaping and reordering: Combine one dimension along with the files or split a dimension into multiple ones; Sort dimensions with associated values.	merge_across_dims merge_across_dims_narm split_multiselected_dims *_depends *_across *_reorder
Dimension and path definition: Identify the key information required to locate files, homogenize dimension and variable names among datasets, and retrieve metadata and ancillary data.	pattern_dims metadata_dims path_glob_permissive return_vars synonims largest_dims_length *_var
File format and data selector support: Specify interface functions for desired file format and conversion between different selector (i.e., data of interest) types	file_opener file_var_reader file_dim_reader file_data_reader file_closer selector_checker

C. Workflow

After the data sources are declared, the next step is to define the operation to be applied. A self-defined function in R function form and the startR functions `Step` and `AddStep` are required to build up the workflow. The workflow adopts the multiApply mechanism, which is an R package and an extension of the base apply function in R. In the apply() fashion, the function only operates on the essential dimensions but not the whole data set, and the operation will loop over the margin dimensions behind the scene. The apply family typically applies a function to a single argument, whereas multiApply efficiently takes one or a list of multi-dimensional arrays as input, suitable for the operation of the declared data arrays.

In Step(), the dimensions to be performed on and the expected returned dimensions are specified by their dimension names, which are more semantically meaningful than the indices. By this means, users can save effort on checking the dimension order and prevent the mistakes hard to detect. After the step is defined, the data and the step are ready to be bundled together as a workflow by AddStep().

D. Data processing

Once the workflow is defined, the final step is to choose the platform to run the execution and specify the execution parameters by the function `Compute`. The platform can be either the workstation that runs the code (locally) or the HPCs that share the same file system with the workstation (remotely).

One important feature of startR, MapReduce paradigm (i.e., chunking), can be realized at this step. Users can specify which dimensions to split the data array along and the number of chunks to make for each, making each chunk fit in the RAM memory. Therefore, even if the total data size is larger than the available RAM, the execution won't overload and crash the platform. The operation defined in the workflow will be applied to each chunk, and the results will be stored in a temporary file. When all the chunks are finished, Compute() will collect and merge the results and return an object including one or multiple multidimensional data arrays as defined in Step(), and the additional metadata generated during the execution.

Since startR features in a multi-node and multi-core manner, the number of execution threads to use for data retrieval and for computation can be determined by the users. If the execution is on a remote HPC, the configuration of the machine needs to be specified in Compute(), including the number of cores per job, the amount of memory to book for each job, the type of workload used by the HPC, and the maximal wall-clock time, etc. Understanding the properties of the machine can help optimize execution efficiency. Besides, the EC-Flow server is required to be installed for job dispatch on HPCs. Through the EC-Flow graphical user interface, users can check and control the execution status of each chunk during the computation. If one chunk fails, it can be re-submitted individually without running the whole execution again, so the risk of time and resource waste is reduced.

E. Summary and future work

startR provides a powerful framework for large multi-dimensional data retrieval and processing. With its clear workflow, a piece of data analysis work can be defined with only tens of lines, and the script is easy to reuse and adapt to other analyses. Though startR has had much flexibility for data structure and operations, it can be strengthened further. For example, only one step (i.e., one function) in the workflow is allowed now. Even though several procedures can be wrapped in one step, multiple steps can optimize the target dimension choice for different operations. Besides, as the popularity of cloud databases grows, retrieving data from cloud-based data systems is worth the development.

Several functionalities in startR, like spatial interpolation and time manipulation, are tailored for earth sciences research, with a special focus on atmospheric sciences such as climate, weather, and air quality. It is compatible with other R tools developed in BSC-CNS, forming a strong toolset for climate research. However, it is potentially competent in other research fields. With the plug-in of other interface functions, startR can be exploited in different scientific domains where large multi-dimensional data is involved.

Author biography

Núria Pérez-Zanón is a postdoctoral researcher and member of the Data and Diagnostic teams in the Computational Earth Sciences group at BSC. With a background in Physics and Meteorology (degrees from the University of Barcelona; UB), she obtained her PhD from Rovira i Virgili University (URV) in 2017 on Climate variability and Change detection in the central Pyrenees using instrumental and paleoclimate proxy data. During the thesis, one year and a half of research stay in LOCEAN (Laboratoire d'Océanographie et du Climat). She is the author of 14 peer-reviewed articles in international journals. She is currently coordinating the development and maintenance of the R tools of the department.