Response to reviewers article 4780

cubble: An R Package for Organizing and Wrangling Multivariate Spatio-temporal Data

H. Sherry Zhang, Dianne Cook, Ursula Laa, Nicolas Langrené, Patricia Menéndez

2023-03-04

Response to B-review_1_1

The Paper

The package and the proposed data structures could certainly constitute a valuable base for the spatio-temporal statistics tooling. Unfortunately the paper does not clearly describe the data structure, nor the associated functionality. I had to read the package vignettes, run the attached code and even look into the package's code in order to fully understand the data structure.

Re: We thank the reviewer for the support on our proposed data structure and the suggestion to improve the paper and the package documentation.

Given the small size of the package, the paper feels a bit on the heavy side. Most of the code in the paper is not self-contained which makes it rather difficult to follow. Examples at the end of the paper don't elucidate the inner workings of the package but rather advertise the functionality of other, mostly visualization, packages.

Re: We select the code and output relevant to demonstrate the cubble package in the examples while leaving the rest in the github repository, where users can look up for more details: https://github.com/huizezhang-sherry/paper-cubble. [on the visualisation packages: xxx]. After addressing all the issues raised by the reviewers, the paper is now xxx pages, including 4 pages of reference and required affiliation.

More fundamentally, the design of the package has an immediately apparent drawback. The cubble class is used to represent two incompatible data types - long and nested. Internally, the two data types are distinguished by the internal attribute "form". In other words, the authors use an attribute of an object to mimic the functionality of a sub-class. A more natural implementation would be start with two distinct data structures, both sub-classes of the abstract cubble_df class, c("spatial_cubble_df", "cubble_df") and c("spatial_temporal_df", "cubble_df"). Such a generic implementation would allow for natural extension of the functionality trough dedicated methods.

Re: We thanks reviewer for the suggestion on the refactorand now:

- **a nested/spatial cubble has class c("spatial_cubble_df", "cubble_df")**
- **a long/temporal cubble has class c("temporal_cubble_df", "cubble_df")**

As a result, S3 methods are now written separately for the two classes unless it doesn't differentiate between the form. For example:

- **face_temporal() will do different things depending on whether it is a nested/long cubble:**

```
face_temporal <- function(...){
  UseMethod("face_temporal")
}

face_temporal.spatial_cubble_df <- function(...){
```

```
}

face_temporal.temporal_cubble_df <- function(...){
  # if already in the temporal face, return the cubble as it is
}
```

- **key_vars() will anyway extract the key variable of a cubble object despite of the form:**

```
key_vars.cubble_df <- function(){
  ...
}
```

Currently an ever-present check on "form" attribute is necessary throughout the code base. To make matters worse, as_cubble currently produces yet another data structure - a list of paired matches, as can be seen from example 5.1 in the paper.

Re: as_cubble() now can coerce four types of foreign objects into a cubble object: 1) a single joint table, 2) NetCDF data, 3) a stars object, 4) a sftime object. The code related to output matching pairs has been moved to a separate function check_key() and reflected in the current example 5.1

Relatedly, the rationale for the parallel naming convention long/nested and temporal/spatial is not entirely clear. It seems to me that the semantically unambiguous temporal/spatial could be used throughout without ambiguity, thus resolving the terminological redundancy.

Re: The naming of long/nested comes from the tidy data concept where the "spatial" form has the nested temporal information and the "temporal" form uses the long form data from the tidy data. We understand it could be more intuitive to refer to them as spaital/ temporal and we add them as a "common name" to the form in the definitional section (now section 2). This naming is also used in the paper when it is more straightforward to use spatial/temporal than nested/long.

My recommendation would be a combination of the following:

- Rework the definitional parts (sections 2 and 3) by following more closely the "design" vignette. **Re: Section 2 and 3 have now been combined into a section 2 and the following subsections are includeded:**

  2.1 introduce the cubble class

  2.2 create and coerce foreign objects into a cubble object

  2.3 functions and methods in cubble (dplyr and cubble functions)

  2.4 compatibility with tsibble and sf

  2.5 comparision with other existing spatio-temporal class: stars and sftime

- Describe at a glance the core functions which operate on the class and their motivation. **Re: A method at glance has been printed at the beginning of subsection 2.3**
- Rework section 4 by making it more concise and provide links to sections in supplementary material or dedicated vignettes. **Re:**
- Rework examples section 5 into "Applications". Briefly describe the applications and provide and refer to dedicated and self-contained vignettes. **Re: xxx**

The Package

The package is generally badly documented. The meaning of the arguments is almost never clear from the documentation alone. For example the documentation of as_cubble states:

Re: Documentation of the package has been improved with a focus on clarifying the input arguemnt types, providing examples, [more]

- What objects are supported as input data? **Re: the objects supported are 1) list with two elements named spatial and temporal, 2) data frame/ tibble, 3) netcdf objects, 4) stars objects, and 5) sftime objects.**
- What is the assumption of the "nestedness" of the input data. **Re:**
- What is the accepted data type of key, index and coords? **Re: Key supports character and factor. Index supports base R classes Date, POSIXlt, POSIXct and tsibble's yearmonth, yearweek, and yearquarter class. Coords support both projected/ unprojected coordinates in columns (can be omitted if created from sf and its subclass).**

Further suggestion

Consider not throwing an error when face_spatial is applied to the spatial object. This would allow for generic code where the "form" of the input the data is not known or does not matter.

Re: Adopted. Now when applying face_spatial to a nested cubble, the cubble will be printed as it is with a message signalling it is already in the nested form. see the reprex below. Same for face_temporal on a long cubble object

```
library(cubble)
nested <- climate_flat |>
  as_cubble(key = id, index = date, coords = c(long, lat))
class(nested)
#> [1] "cubble_df"  "rowwise_df" "tbl_df"     "tbl"        "data.frame"
face_spatial(nested)
#> i The cubble is already in the nested form
#> # cubble:   id [5]: nested form
#> # bbox:     [115.9764, -32.9342, 133.5407, -12.4239]
#> # temporal: date [date], prcp [dbl], tmax [dbl], tmin [dbl]
#>   id            lat  long  elev name           wmo_id ts
#>   <chr>       <dbl> <dbl> <dbl> <chr>           <dbl> <list>
#> 1 ASN00009021 -31.9  116.  15.4 perth airport   94610 <tibble [366 × 4]>
#> 2 ASN00010311 -31.9  117. 179   york            94623 <tibble [366 × 4]>
#> 3 ASN00010614 -32.9  117. 338   narrogin        94627 <tibble [366 × 4]>
#> 4 ASN00014015 -12.4  131.  30.4 darwin airport  94120 <tibble [366 × 4]>
#> 5 ASN00015131 -17.6  134. 220   elliott         94236 <tibble [366 × 4]>
```

Created on 2023-05-22 with reprex v2.0.2

It would be useful to print a few rows of the spatial attribute of the "long" object.

Re: the spatial attribute can be checked with function spaital(<cubbble_obj>)

Section 3.6 is missing `tidyr` package which is especially relevant here given its nesting and unnesting operations https://tidyr.tidyverse.org/articles/nest.html

Re: xxx

More comments inlined in the pdf.

- on p4 "The following sections explain their roles, why the new cubble class is needed and how the package relates to existing packages for spatial and temporal data analysis." The paper is a bit on the verbose side. Such comments could be safely removed. **Re: sentence removed.**
- on p5 section 3.1, "The arguments key and index follow the wording in the tsibble package to describe the temporal order and multiple series while coords specifies the spatial location of each site.", Unclear what `key` and `index` mean. Isn't key a spatial identifier and index a temporal one? **Re: [TODO] Yes,**

key is the spatial identifier and index is the temporal one. They are now explained in the text as well as referenced to the tsibble package.

- on p5 section 3.1, the word "wording" is highlighted in the sentence "The arguments key and index follow the wording in the tsibble package to . . . ", conventions. **Re: *wording* changed to *conventions***
- on p6 section 3.2, 'The cubble class also provides a long form, which expands the ts column and temporarily "hides" the spatial variables.', confusing: `hides` means storing it as an attribute. **Re: The sentence is now changed to ..., which expands the ts column and stores the spatial variables as an attribute**
- on p6 section 3.2, I think it would be a good idea to clearly describe the two data structures in an earlier section. By this point it's still not clear what those are and what the meaning of key and index is. **Re: addressed in bullet point "on p5 section 3.1"**
- on p6, section 3.2, yet another reason to call this data structure "cubble_temporal" instead of the semantically unobvious "long". **Re: see paragraph 4 under section The paper**
- on p8 section 3.5, Isn't the root reason the memory efficiency? That is, the spatial component cannot be represented as part of the nested tibble without repetition. **Re: this has now been incorporated in subsection 2.5 to compare with a sftime object**
- on p8 section 3.6, The missing elephant is tidyr which has nesting and unnesting operations https://tidyr.tidyverse.org/articles/nest.html. **Re: addressed under section Further suggestion in this response**

Response to B-review_2_1

The Manuscript

The manuscript is in general clearly structured. However, an important reference to the R-package sftime is missing that also deals with the representation of spatiotemporal data. It might be beneficial to provide illustrative examples in the manuscript that clearly compare between a cubble representation and representations of existing packages. Which data sets/structures cannot (or with a greater effort) be represented with existing R-packages such as e.g. stars and sftime?

Re: The comparison with other existing spatio-temporal class, i.e. stars and sftime, has been addressed in subsection 2.5 (after combining the original section 2 and 3 into section 2)

For spatial data, the coordinate reference system is essential metadata information. In one example of the manuscript, the input data does have information on the CRS but it is not discussed in the manuscript how that is handled within cubble. How would coordinate transformations be handled with cubble?

Re: We understand spatial data is, strictly speaking, not uniquely identified if CRS information is not provided, while many wild data still do not contain this crucial piece of information. In cubble, if the spatial data is from an sf object, the cubble will retain the sf class and users are welcomed to use functions in sf to perform coordinate transformations. If not, the function cubble::add_geometry_column() can be used to construct the geometry column from long and lat, and the resulting cubble will have the sf class. if CRS is missing, cubble::add_geometry_column() will by default assume OGC:CRS84 (WGS84).

The use-case of temporal matching based on features of the time series appears an interesting, but also a very specific one. How could this be generalized to a more generic approach?

Re: xxx

Additionally, several typos and language issues arise and limit the readability of the manuscript. Some examples are:

- doubled/missing words/none correct sentences:

  - "*components* spatio-temporal *components*" **Re: to "... spatio-temporal components"**
  - "... fits works ..." **Re: to ".. works well with ..."**
  - "... be activate rows ..." **Re: to "... will activate rows ..."**
  - "... highlighted *in* the ..." **Re: to "... highlighted on the map"**
  - ". . . using 2020 measurements using match_sites() function." **Re: to "... using the match_sites() function"**
  - "An example of this using is included in the Appendix" **Re: to "... this use ..."**
  - ". . . the data in -a- multiple -of- ways on-the-fly" **Re: to "... in multiple ways ..."**

- Surprising references: in Section 4.4 it says ". . . Glyph maps (Section 3.4)", but Section 3.4 says ". . . glyph maps will be explained in Section 4.4". **Re: the looped reference in section 3.3 and 3.4 is replaced with (Wickham et al. 2012).**

- Should ". . . it's temperatures are more *consistent*" rather be ". . . it's temperatures are more *constant*"? **Re: to "more constant"**

- Typos:

  - ". . . spatial and *tmeporal* information are available." **Re: to "temporal"**
  - The polar vortex, signalled by the high *speicfic* humidity, splits into two on 2002-09-26 and *further-s- split_s_* into four on 2002-10-04. **Re: to "specific", "further splits"**
  - the data in *-a- multiple -of-* ways on-the-fly. **Re: to "multiple ways"**

The package

I have been a bit puzzled by the "print" of a nested cubble in its temporal face. The row # temporal: date [date], prcp [dbl], tmax [dbl], tmin [dbl] appears at first sight (also in comparison with the print of the spatial face), as if the temporal domain is given by all those variables. To me, a notion such as

```
# temporal: date [date]
```

```
# variables: prcp [dbl], tmax [dbl], tmin [dbl]
```

would have been more intuitive. Possibly also including the temporal range (as for the spatial domain its bbox).

Re: Thanks reviewers for the suggestion and this has now been incorporated in the package. The second line in the header, when printed in the long (temporal) form will now show the start and the end date, the date interval, and whether there are gaps in the data. See the reprex example below:

```
library(cubble)
face_temporal(climate_aus)
#> # cubble:  key: id [639], index: date, long form
#> # extent:  2020-01-01 -- 2020-12-31 [1D], has gaps!
#> # spatial: lat [dbl], long [dbl], elev [dbl], name [chr], wmo_id [dbl]
#>    id          date        prcp  tmax  tmin
#>    <chr>       <date>     <dbl> <dbl> <dbl>
#>  1 ASN00001006 2020-01-01   164  38.3  25.3
#>  2 ASN00001006 2020-01-02     0  40.6  30.5
#>  3 ASN00001006 2020-01-03    16  39.7  27.2
#>  4 ASN00001006 2020-01-04     0  38.2  27.3
#>  5 ASN00001006 2020-01-05     2  39.3  26.7
#>  6 ASN00001006 2020-01-06    60  32.9  25.6
#>  7 ASN00001006 2020-01-07   146  34.1  25.5
#>  8 ASN00001006 2020-01-08    40  36.6  26.2
#>  9 ASN00001006 2020-01-09     0  38.2  27.6
```

```
#> 10 ASN00001006 2020-01-10     0  38.9  29.7
#> # ... with 230,750 more rows
```

Created on 2023-05-23 with reprex v2.0.2

In the manuscript and in the reproducible-script.R, a data set `historical_tmax` is introduced. In the package, the corresponding data set seems to be `tmax_hist`.

Re: [TODO] The two datasets are related but not the same. The data historical_tmax contains daily weather stations data for the two periods: 1971-1975 and 2016-2020 and amounts to 3.2MB, which is not suitable to include in the package given its size. The tmax_hist summarises the daily data into monthly to produce the dataset suitable to include in the package. The documentation to tmax_hist now include more details on this.

The manuscript cannot be reproduced, as the reproducible-script.R (part of the submission to JSS) only uses a subset, but does not provide a separate result file that would allow to compare the reproduced outputs with the desired output for the subset.

Re: A result file in html has been submitted this time.