

Package: clim4health (via r-universe)

June 30, 2026

Title Post-Processing of Climate Data for Health Applications

Version 0.1.0

Description Obtain, transform and export climate data including reanalyses, (seasonal) forecasts and hindcasts, and weather stations for their use in epidemiological analyses. It is organised in three sequential blocks, input (download and load data), transform (downscaling, verification, spatiotemporal aggregation and threshold-based indicators) and output (visualising and exporting data). Downscaling methods include those described in Duzenli et al. (2026) [doi:10.1038/s41598-026-45067-2](https://doi.org/10.1038/s41598-026-45067-2) and verification methods are based on those in Manubens et al. (2018) [doi:10.1016/j.envsoft.2018.01.018](https://doi.org/10.1016/j.envsoft.2018.01.018).

License AGPL (>= 3)

URL <https://gitlab.earth.bsc.es/ghr/clim4health>,
<https://bsc-es.github.io/GHRtools/docs/clim4health/clim4health>

BugReports <https://gitlab.earth.bsc.es/ghr/clim4health/-/issues>

Encoding UTF-8

Imports CStools, ecmwfr, ggplot2, GHRexplore, lubridate, s2dv, sf, stars, terra, data.table, units, CSDownscale, ncdf4, ggpattern, startR, rlang

Suggests testthat (>= 3.0.0), spData, cubelyr, knitr, withr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

SystemRequirements CDO (Climate Data Operators, <https://code.mpimet.mpg.de/projects/cdo>), NCO (NetCDF Operators, <https://nco.sourceforge.net/>)

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

NeedsCompilation no

Author Emily Ball [aut] (ORCID: <https://orcid.org/0000-0002-3002-4068>), Carles Milà [aut, cre] (ORCID: <https://orcid.org/0000-0003-0470-0760>), Alba Llabrés [aut] (ORCID: <https://orcid.org/0000-0003-2144-675X>), Raul Capellan [aut], Rebeca Nunes [aut] (ORCID: <https://orcid.org/0009-0009-8738-0985>), Daniela Lührsen [aut] (ORCID: <https://orcid.org/0009-0002-6340-5964>), Anna B. Kawiecki [aut] (ORCID: <https://orcid.org/0000-0002-0499-2612>), Rachel Lowe [aut] (ORCID: <https://orcid.org/0000-0003-3939-7343>)

Maintainer Carles Milà <carles.milagarcia@bsc.es>

Config/pak/sysreqs libnetcdf-dev

Repository <https://cran.r-universe.dev>

Date/Publication 2026-06-30 21:10:20 UTC

RemoteUrl <https://github.com/cran/clim4health>

RemoteRef HEAD

RemoteSha 7a6ae26904d7a6aaa94d4939e23fd0b27ed2a181

Contents

c4h_collapse	2
c4h_convert	4
c4h_convert_units	5
c4h_data	6
c4h_downscale	7
c4h_get	10
c4h_get_help	13
c4h_index	13
c4h_load	15
c4h_plot	17
c4h_plotskill	19
c4h_save	22
c4h_space	23
c4h_time	24
c4h_verify	25

Index **31**

c4h_collapse	<i>Collapse one dimension of a s2dv_cube object</i>
--------------	---

Description

Collapse one dimension of the *s2dv_cube*. `c4h_collapse()` works differently depending on the dimension being collapsed:

- Use `dim = "ensemble"` to compute a summary statistic across ensemble members in forecast and hindcast datasets.
- Use `dim = "time"` to compute a time aggregation with observational data (i.e. the length of the 'sdate' dimension must be one). In this case the first "time" of the dataset is assigned as the new *s2dv* time coordinate. To compute time aggregations when 'sdate' has length greater than one, use the function `c4h_time()` instead.
- Use `dim = "sdate"` to compute a climatology, if each 'sdate' corresponds to a different year. In this case the first "sdate" of the dataset is assigned as the new *s2dv* sdate coordinate.

Usage

```
c4h_collapse(data, dim, fun = "mean", na.rm = TRUE)
```

Arguments

<code>data</code>	<i>s2dv_cube</i> object with the climate data.
<code>dim</code>	Dimension to collapse. Options include 'ensemble', 'sdate' and 'time'.
<code>fun</code>	Function to perform the aggregation, defaults to 'mean'. Other options include 'median', 'min' and 'max'.
<code>na.rm</code>	Logical, indicating whether NA values should be ignored when applying the collapsing function. Defaults to TRUE.

Value

The collapsed *s2dv_cube*.

Examples

```
# Load a forecast
fcst <- c4h_load(system.file("extdata", "forecast", package = "clim4health"),
                var = "t2m",
                year = 2025,
                month = 1,
                ext = 'nc')

fcst$dims

# Ensemble median of a forecast
fcst_coll <- c4h_collapse(fcst, dim = "ensemble", fun = "median")
fcst_coll$dims

# Load an observational time-series dataset
stations_path <- system.file("extdata", "stations", package = "clim4health")
stations <- c4h_load(stations_path, var = "temp_mean", ext = "csv")
stations$dims
```

```
# Compute time-mean (average temp conditions)
stations_coll <- c4h_collapse(stations, dim = "time")
stations_coll$dims

# Load an observational dataset aligned to a hindcast
obs_path <- system.file("extdata", "reanalysis", package = "clim4health")
obs <- c4h_load(obs_path, var = "t2m", ext = "nc",
               year = 2010:2012, month = 1, leadtime_month = 1:3)
obs$dims

# Compute climatology (average over years)
obs_coll <- c4h_collapse(obs, dim = "sdate")
obs_coll$dims
```

c4h_convert

c4h_convert

Description

Convert *s2dv_cube* objects to *terra*, *stars*, *data.frame*, and *sf*.

Usage

```
c4h_convert(data, output_format, crs = NULL, drop = FALSE)
```

Arguments

<code>data</code>	The object to transform.
<code>output_format</code>	A string specifying the desired output format. Valid options are 'terra' (SpatRaster), 'stars', 'data.frame' or 'sf' (polygons).
<code>crs</code>	Coordinate reference system to use for output_formats 'terra' and 'sf'. The default is NULL, in which case the CRS will be taken from the input data if available, or assumed to be WGS 84 (EPSG:4326) for gridded data.
<code>drop</code>	Logical. If TRUE, dimensions with only one value will be dropped when converting to the desired format. Default is FALSE.

Details

The function only allows to convert into the following classes depending on the data format.

Gridded data 'terra' SpatRaster, 'stars', 'sf' polygons, or 'data.frame'.

Areal data 'sf' polygons or 'data.frame'.

Point data 'data.frame'.

Value

An R object of type *terra* SpatRaster, *stars*, *data.frame* or *sf* polygons.

Examples

```

# Load example forecast data
fcst_path <- system.file("extdata", "forecast", package = "clim4health")
fcst <- c4h_load(fcst_path, variable = "t2m", leadtime_month = 1, ext = 'nc')

#####
### Convert gridded data
## to terra
out <- c4h_convert(fcst, "terra", crs = "EPSG:4326")
## to data.frame
out <- c4h_convert(fcst, "data.frame")
## to sf
out <- c4h_convert(fcst, "sf", crs = "EPSG:4326")

#####
#### load polygon data ####
library(sf)
# Region municipalities
municip_path <- system.file("extdata", "areas", "municip_vallecauca.gpkg",
                           package = "clim4health")
municip <- read_sf(municip_path)
municip <- municip[1:2, ]
fcst_aggr <- c4h_space(fcst, municip)

### Convert polygon data
## to data.frame
out <- c4h_convert(fcst_aggr, "data.frame")
## to sf
out <- c4h_convert(fcst_aggr, "sf", crs = "EPSG:4326")

#####
#### load station data ####
station_path <- system.file("extdata", "stations", "temp_vallecauca.csv",
                           package = "clim4health")
stations <- c4h_load(station_path, variable = "temp_mean", ext = 'csv')

### Convert location data
## to data.frame
out <- c4h_convert(stations, "data.frame")

```

c4h_convert_units

c4h_convert_units

Description

Convert the units of a variable or variables in an `s2dv_cube`. If the `'to'` argument is `NULL`, the current units will be displayed instead of performing any conversion.

Usage

```
c4h_convert_units(data, to = NULL, var = NULL, from = NULL)
```

Arguments

data	An <i>s2dv_cube</i> object containing the data to be converted.
to	A character string specifying the target units to convert to. If NULL, no conversion will be performed and the current units will be displayed. The default is NULL.
var	A character string specifying the variable to convert. If NULL, all variables will be converted. The default is NULL.
from	A character string specifying the current units of the variable. If NULL, the current units will be inferred from the data. The default is NULL.

Value

An *s2dv_cube* object with the converted units.

Examples

```
# Forecast data included in the package
fcst_path <- system.file("extdata/forecast/", package = "clim4health")
fcst_path <- paste0(fcst_path, "/")

# Load the forecast data
fcst <- c4h_load(fcst_path,
                var = "t2m",
                year = 2025,
                month = 1,
                ext = 'nc')

# Display the current units
c4h_convert_units(fcst, var = "t2m")

# Convert from Kelvin to Celsius
fcst_celsius <- c4h_convert_units(fcst,
                                  to = "celsius",
                                  var = "t2m",
                                  from = "K")
```

c4h_data

Example datasets included in clim4health

Description

These datasets are stored in `inst/extdata/` and are accessed through functions such as `c4h_load` or via `system.file("extdata", "file.csv", package = "clim4health")`.

Station and administrative data

- `stations/temp_vallecauca.csv`: Table data containing daily temperature station observations for the *Valle del Cauca* department in Colombia (2010-2012). The raw hourly data have been aggregated to daily measurements whenever at least a 70% of the hourly observations were available for a given day. Source: IDEAM (2025).
- `areas/munip_vallecauca.gpkg`: Geopackage containing the municipality boundaries of the "Valle del Cauca" department in Colombia. Source: IGAC (2025).

Climate forecasts

- `forecast/t2m_20250101.nc`: 3-month monthly-averaged seasonal forecast (ECMWF SEAS5.1 seasonal forecast) including 2-metre temperature (K, 51 ensemble members, 1° spatial resolution) issued in January 2025. The spatial extent covers the "Valle del Cauca" department in Colombia.

Climate hindcasts

- `hindcast/t2m_*.nc`: 3-month monthly-averaged seasonal hindcast (ECMWF SEAS5.1 seasonal forecast) including 2-metre temperature (K, 25 ensemble members, 1° spatial resolution) issued in January 2010-2012. The spatial extent covers the *Valle del Cauca* department in Colombia.

Climate reanalysis

- `reanalysis/t2m_*.nc`: monthly-averaged ERA5-Land reanalysis product including 2-metre temperature (K, 0.1° spatial resolution) for the months of January, February and March 2010-2012. The spatial extent covers the *Valle del Cauca* department in Colombia.

See Also

[c4h_load](#)

c4h_downscale

c4h_downscale

Description

Downscale and calibrate climate data.

Usage

```
c4h_downscale(  
  downscale_function,  
  exp,  
  obs = NULL,  
  exp_cor = NULL,  
  bbox = NULL,
```

```

ncores = NULL,
target_grid = NULL,
method_bc = NULL,
method_remap = NULL,
points = NULL,
method_point_interp = NULL,
method_lr = NULL,
loocv = TRUE,
...
)

```

Arguments

downscale_function	The downscaling function to use. Options are: "Interpolation", "Intbc", "Intlr", "Analog".
exp	The experimental data to downscale. It must be an <i>s2dv_cube</i> .
obs	The observational data to downscale. It must be an <i>s2dv_cube</i> .
exp_cor	Optional data to downscale. It must be an <i>s2dv_cube</i> . If provided, it will be downscaled using exp and obs; if not provided, the exp will be downscaled instead. The default value is NULL.
bbox	A numeric vector indicating the borders of the region defined in obs. It consists of four elements in this order: North, West, South, East. If set to NULL (default), the function <code>c4h_downscale</code> will take the minimum and maximum values of the latitudes and longitudes.
ncores	An integer indicating the number of cores to use in parallel computation. The default value is NULL.
target_grid	A character vector indicating the target grid (in the case of Interpolation techniques) or the coarse grid (in the case of Analogs) to be passed to CDO. It must be a grid recognised by CDO or a NetCDF file. The default value is NULL. If NULL, the grid of obs will be used as the target grid for Interpolation techniques, and the grid of exp will be used as the coarse grid for Analogs.
method_bc	A character vector indicating the bias adjustment method to be applied after an interpolation. Accepted methods are 'quantile_mapping', 'dynamical_bias', 'bias', 'evmos', 'mse_min', 'crps_min', 'rpc-based'. The abbreviations 'dbc', 'qm' can also be used. The default value is NULL.
method_remap	A character vector indicating the regridding method to be passed to CDORemap. Accepted methods are "con", "bil" (or "bilinear"), "bic", "nn", "con2". If "nn" method is to be used, CDO_1.9.8 or newer version is required. The default value is NULL.
points	A list of two elements containing the point latitudes and longitudes of the locations to downscale the model data. The list must contain the two elements named as 'latitude' and 'longitude'. If the downscaling is to a point location, only regular grids are allowed for exp and obs. Only needed if the downscaling is to a point location.

method_point_interp	A character vector indicating the interpolation method to interpolate model gridded data into the point locations. Accepted methods are "nearest", "bilinear", "9point", "invdist4nn". Only needed if the downscaling is to a point location.
method_lr	A character vector indicating the linear regression method to be applied. Accepted methods are 'basic' and '9nn'. The 'basic' method fits a linear regression using high resolution observations as predictands and the interpolated model data as predictor. Then, the regression equation is applied to the interpolated model data to correct the interpolated values. The '9nn' method uses a linear regression with the nine nearest neighbours as predictors and high-resolution observations as predictands. Instead of constructing a regression model using all nine predictors, principal component analysis is applied to the data of neighboring grids to reduce the dimension of the predictors. The linear regression model is then built using the principal components that explain 95% of the variance. The '9nn' method does not require a pre-interpolation process. The default value is NULL.
loocv	A logical indicating whether to apply leave-one-out cross-validation when applying the bias correction. The default is TRUE. In this case, when applying the bias correction or building the regression model for a given year, the corresponding values from that year are removed.
...	Additional arguments to be passed to the CSDownscale function.

Value

A list of statistically downscaled data as *s2dv_cube* objects.

Note

See `clim4health_downscaling.Rmd` in the package vignettes for a detailed tutorial on function arguments and usage.

Examples

```
# DISCLAIMER: The purpose of these examples is to show the functionality
# of the different functions. To perform downscaling and calibration, a
# much larger time series is required.
hcst <- c4h_load(system.file("extdata", "hindcast", package = "clim4health"),
                variable = "t2m",
                year = 2010:2012,
                month = 1,
                leadtime_month = 1:3,
                ext = "nc")
lats <- hcst$coords$latitude
lons <- hcst$coords$longitude
obs <- c4h_load(system.file("extdata", "reanalysis",
                           package = "clim4health"),
               variable = "t2m",
               year = 2010:2012,
               month = 1,
               leadtime_month = 1:3,
```

```

        bbox = c(max(lats), min(lons),
                 min(lats), max(lons)),
        ext = "nc")
### Example with gridded data
dwn <- c4h_downscale("Interpolation", exp = hcst,
                    method_remap = "bilinear",
                    target_grid = system.file("extdata",
                                              "reanalysis/t2m_201001.nc",
                                              package = "clim4health"))

## Example with polygon data
shp_path <- system.file("extdata", "areas", "munip_vallecauca.gpkg",
                       package = "clim4health")
obs_agg <- c4h_space(obs,
                    areas = shp_path,
                    fun = "mean",
                    areas_id = "munip_code")
hcst_agg <- c4h_space(hcst,
                    areas = shp_path,
                    fun = "mean",
                    areas_id = "munip_code")
hcst_agg_cal <- c4h_downscale(downscale_function = "Intbc",
                             exp = hcst_agg,
                             obs = obs_agg,
                             method_bc = "evmos")

## Example with station data
point_locs <- clim4health::c4h_load(system.file("extdata",
                                              "stations",
                                              package = "clim4health"),
                                  ext = "csv",
                                  year = 2010:2012,
                                  month = 1,
                                  leadtime_month = 1:3,
                                  variable = "temp_mean")
points <- list(latitude = point_locs$attrs$location$latitude,
              longitude = point_locs$attrs$location$longitude)
point_locs <- c4h_time(point_locs,
                      time_aggregation = "monthly",
                      dim_aggregation = "sdate",
                      fun = "mean")

dwn <- c4h_downscale("Interpolation", exp = hcst,
                    points = points, method_point_interp = "bilinear")

```

c4h_get

c4h_get

Description

Download netCDF climate data from the Copernicus Climate Data Store.

Usage

```

c4h_get(
  pat,
  dataset,
  product_type,
  variable,
  year,
  month = 1:12,
  day = 1:31,
  time = 0,
  bbox,
  leadtime_month,
  originating_centre,
  system,
  outname = paste0("cds"),
  outpath = getwd()
)

```

Arguments

pat	Your personal access token (PAT). You can find your PAT in your CDS profile after logging in.
dataset	The name of the dataset you want to load. Available datasets can be found via c4h_get_help .
product_type	The product type that you want to load. Available variables can be found via c4h_get_help .
variable	The variable that you want to load. Available variables can be found via c4h_get_help .
year	A vector of years, starting in 1950.
month	A vector of months. e.g. c(1,2) for January and February. Default is all months.
day	A vector of days in the month. Default is all days.
time	A vector of hours (0-23) in UTC time, only needed when datasets are hourly. Note that you may need to convert your local time zone to UTC to get the correct time points. Default is 0 (midnight UTC).
bbox	A vector of coordinates. (in order: North, West, South, East).
leadtime_month	A vector of months for the forecast leadtime. e.g. c(1,2) for January and February. Default is all months.
originating_centre	The originating centre of the forecast if parameter dataset is "seasonal-monthly-single-levels". Options are "ecmwf", "ukmo", "meteo_france", "dwd", "cmcc", "ncep", "jma", "eccc", and "bom". There is no default value for this parameter.
system	Model identifier of the forecast if parameter dataset is "seasonal-monthly-single-levels". Options are 1, 2, 3, 4, 5, 6, 7, 8, 9, 12, 13, 14, 15, 21, 22, 35, 51, 600, 601, 602, 603, 604, 605, 610. Valid options depend on the value of originating_centre. There is no default value for this parameter.
outname	The name stem of the resultant netcdf file.

outpath The path where the resultant netcdf file should be saved. Default is the current working directory.

Value

A netcdf file or files.

See Also

[c4h_get_help](#) to get help about the available datasets and parameters used in c4h_get.

Examples

```
## Not run:
# Download ERA5 land 2m temperature reanalysis data
c4h_get(pat = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
        dataset = "reanalysis-era5-land-monthly-means",
        product_type = "monthly_averaged_reanalysis",
        variable = "2m_temperature",
        year = c(2010, 2011, 2012),
        month = c(4, 5),
        bbox = c(33, -93, -23, -17),
        outname = "era5land")

# Download 2m temperature system 51 ECMWF monthly seasonal forecasts
# to be used as forecasts and hindcasts respectively.
c4h_get(pat = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
        dataset = "seasonal-monthly-single-levels",
        originating_centre = c("ecmwf"),
        system = c("51"),
        variable = c("2m_temperature"),
        product_type = c("monthly_mean"),
        year = c(2010, 2011, 2012),
        month = 4,
        leadtime_month = 1:3,
        bbox = c(33, -93, -23, -17),
        outname = "hindcast")

c4h_get(pat = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
        dataset = "seasonal-monthly-single-levels",
        originating_centre = c("ecmwf"),
        system = c("51"),
        variable = c("2m_temperature"),
        product_type = c("monthly_mean"),
        year = 2024,
        month = 4,
        leadtime_month = 1:3,
        bbox = c(33, -93, -23, -17),
        outname = "forecast")

## End(Not run)
```

c4h_get_help	<i>Displays a list of available climate datasets, variables and parameters.</i>
--------------	---

Description

Displays a list of available climate datasets that can be downloaded from the Copernicus Climate Data Store using [c4h_get](#), as well as information about the available variables and parameters.

Usage

```
c4h_get_help(dataset, parameter)
```

Arguments

dataset	If dataset is provided, then the list of required parameters for that dataset will be returned. If no dataset is provided, a list of available datasets will be returned.
parameter	If parameter is provided, a list with all possible options for that parameter is returned. If no parameter is provided then the list of required parameters for the given dataset will be returned.

Value

Prints a list of either available climate datasets, required parameters or options to the console.

See Also

[c4h_get](#) to download climate data.

Examples

```
c4h_get_help()  
c4h_get_help("reanalysis-era5-land")  
c4h_get_help("reanalysis-era5-land", "product_type")  
c4h_get_help("reanalysis-era5-land", "variable")
```

c4h_index	<i>Apply a threshold-based mask to climate data based on value ranges.</i>
-----------	--

Description

Apply a threshold-based mask to climate data based on value ranges.

Usage

```
c4h_index(
  data,
  return_mask = FALSE,
  lower_threshold = NULL,
  upper_threshold = NULL,
  closed = TRUE
)
```

Arguments

<code>data</code>	<i>s2dv_cube</i> object with the climate data to be masked, usually read using <code>c4h_load</code> .
<code>return_mask</code>	Boolean. If TRUE (default), return original values where the condition is fulfilled and NA otherwise. If FALSE, return 1 where the condition is met and 0 otherwise.
<code>lower_threshold</code>	Numeric. Optional lower threshold of the mask. All cells below that value will be masked out.
<code>upper_threshold</code>	Numeric. Optional upper threshold of the mask. All cells above that value will be masked out.
<code>closed</code>	Boolean. If TRUE (default), use closed intervals including the threshold values. If FALSE, intervals are open.

Value

An *s2dv_cube* with the values masked or the mask (depending on `return_mask`).

Examples

```
# Read reanalysis data
obs_path <- system.file("extdata", "reanalysis", package = "clim4health")
obs <- c4h_load(obs_path,
  variable = "t2m",
  year = 2010,
  month = 1,
  ext = "nc")
c4h_plot(obs)

# Compute index, temperature higher than 290K
index <- c4h_index(obs, lower_threshold = 290)
c4h_plot(index)

# Return mask, temperature higher than 290K
mask <- c4h_index(obs, return_mask = TRUE, lower_threshold = 290)
c4h_plot(mask)
```

c4h_load

c4h_load

Description

Load forecast, hindcast, or observational climate data (reanalysis or stations) as an *s2dv_cube* object.

Usage

```
c4h_load(
  path,
  variable = NULL,
  year = "all",
  month = "all",
  day = "all",
  time = "all",
  leadtime_month = "all",
  ext = NULL,
  bbox = NULL
)
```

Arguments

path	String to the folder containing the files to be loaded, or a vector of specific file paths.
variable	The variable that you want to load.
year	A vector of years, starting in 1950.
month	A vector of months. e.g. c(1,2) for January and February. Default is all months.
day	A vector of days in the month. Default is all days.
time	A vector of hours (0-23). Only needed when datasets are hourly. Default is all hours.
leadtime_month	A vector of months for the forecast leadtime. e.g. c(1,2) for January and February for data initialized in January. Default is "all". If "all" for hindcast and forecast data, all leadtime months will be loaded. If "all" for reanalysis or station data, data will be loaded as a time series in the time dimension, and the sdate dimension will be set to 1.
ext	File extension. Options include "nc", ".nc", "csv", and ".csv".
bbox	A vector of coordinates in order: c(lat_max, lon_min, lat_min, lon_max).

Value

An *s2dv_cube* object.

Examples

```
# Forecast data included in the package
fcst_path <- system.file("extdata/forecast/", package = "clim4health")
fcst_path <- paste0(fcst_path, "/")

# Load the forecast data
fcst <- c4h_load(fcst_path,
                variable = "t2m",
                year = 2025,
                month = 1,
                ext = "nc")

class(fcst)
dim(fcst$data)
summary(fcst$data)

# Hindcast data included in the package
hcst_path <- system.file("extdata/hindcast/", package = "clim4health")
hcst_path <- paste0(hcst_path, "/")

# Load the hindcast data
hcst <- c4h_load(hcst_path,
                variable = "t2m",
                year = 2010:2012,
                month = 1,
                leadtime_month = 1:3,
                ext = "nc")

class(hcst)
dim(hcst$data)
summary(hcst$data)

# Reanalysis data included in the package
rean_path <- system.file("extdata/reanalysis/", package = "clim4health")
rean_path <- paste0(rean_path, "/")

# Load the reanalysis data with dates to match the hindcast
rean <- c4h_load(rean_path,
                variable = "t2m",
                year = 2010:2012,
                month = 1,
                leadtime_month = 1:3,
                ext = "nc")

class(rean)
dim(rean$data)
summary(rean$data)

# Load the reanalysis data as a time series
rean <- c4h_load(rean_path,
                variable = "t2m",
                year = 2010,
                month = 1:3,
                leadtime_month = "all",
                ext = "nc")
```

```
class(rean)
dim(rean$data)
summary(rean$data)

# Load station data as a time series matching the reanalysis above
stat_path <- system.file("extdata/stations/", package = "clim4health")
stat_path <- paste0(stat_path, "/")

station <- c4h_load(stat_path,
                   variable = "temp_mean",
                   year = 2010,
                   month = 1:3,
                   leadtime_month = "all",
                   ext = "csv")

class(station)
dim(station$data)
summary(station$data)
```

c4h_plot

Plot s2dv climate data

Description

Plots *s2dv_cube* gridded, polygon and point climate data generated by `clim4health` functions.

Usage

```
c4h_plot(
  data,
  var = NULL,
  sdate = NULL,
  time = NULL,
  ensemble = NULL,
  ensemble_fun = mean,
  boundaries = NULL,
  mask_boundaries = FALSE,
  title = NULL,
  legend = NULL,
  palette = "-RdBu",
  centering = NULL,
  bins = NULL,
  bins_method = "equal",
  bins_label = NULL,
  coordgrid = FALSE,
  ...
)
```

Arguments

<code>data</code>	<code>s2dv_cube</code> object with the climate data to be plotted.
<code>var</code>	Numerical index or name of the variable to be plotted. Only required if more than one variable is available.
<code>sdate</code>	Numerical index of the forecast starting time to be plotted. Defaults to NULL (all starting dates).
<code>time</code>	Numerical index of the time to be plotted. Defaults to NULL (all times).
<code>ensemble</code>	Numerical index of the ensemble member to be plotted. If set to TRUE, it aggregates all the ensemble members. Defaults to NULL (all ensemble members).
<code>ensemble_fun</code>	Aggregation function used when <code>ensemble = TRUE</code> . Defaults to the mean.
<code>boundaries</code>	Polygon boundaries to add to the map. Set to NULL for no boundaries (default), otherwise supply a polygon <code>sf</code> object or set to "countries" for coarse country boundaries.
<code>mask_boundaries</code>	If TRUE (default), mask the data to the provided boundaries.
<code>title</code>	Optional plot title.
<code>legend</code>	Optional legend title. If NULL (default), it uses the variable names.
<code>palette</code>	GHR, RColorBrewer or colorspace palette. Use "-" before the palette name (e.g., "-Reds") to reverse it. Alternatively, a vector of hex codes to create a continuous interpolated palette.
<code>centering</code>	Value at which to center the colour palette. Defaults to NULL (no centering).
<code>bins</code>	Number of bins for categorization of numerical variables. Defaults to NULL (no binning).
<code>bins_method</code>	Method to compute the bins, only used when <code>bins</code> is no NULL. Possible values are "quantile" and "equal" (default).
<code>bins_label</code>	Optional labels for the bins. They must have the same length as the number of bins. Defaults to NULL (default interval labels).
<code>coordgrid</code>	If TRUE, display coordinate grids on the map.
<code>...</code>	Extra arguments indicating fixed aesthetics to be passed to <code>stars::geom_stars</code> , including <code>lwd</code> , <code>alpha</code> and <code>colour</code> .

Value

A `ggplot2` plot with the climate data.

See Also

[c4h_plotskill](#) for plotting forecast skill metrics from [c4h_verify](#).

Examples

```

library("sf")

# Region municipalities
mun_path <- system.file("extdata", "areas", "munip_vallecauca.gpkg",
                        package = "clim4health")
mun <- read_sf(mun_path)

# Forecast
fcst_path <- system.file("extdata", "forecast", package = "clim4health")
fcst <- c4h_load(fcst_path, variable = "t2m", ext = 'nc')
c4h_plot(fcst, title = "Forecast", ensemble = TRUE)

# Reanalysis
rean_path <- system.file("extdata", "reanalysis", package = "clim4health")
rean <- c4h_load(rean_path, variable = "t2m", ext = 'nc')
c4h_plot(rean, title = "Reanalysis", boundaries = mun,
         mask_boundaries = TRUE, coordgrid = TRUE)

# Hindcast
hind_path <- system.file("extdata", "hindcast", package = "clim4health")
hind <- c4h_load(hind_path, variable = "t2m", ext = 'nc')
c4h_plot(hind, title = "Hindcast", ensemble = TRUE)

# Stations
stations_path <- system.file("extdata", "stations", package = "clim4health")
stations <- c4h_load(stations_path, var = "temp_mean", ext = "csv")
c4h_plot(stations, title = "Stations", time = 1:2,
         boundaries = mun, coordgrid = TRUE)

# Spatially aggregated data
fcst_aggr <- c4h_space(fcst, mun)
c4h_plot(fcst_aggr, ensemble = 1:5, lwd = 0.1,
         title = "Temp. by municipality", coordgrid = TRUE)

```

c4h_plotskill

Plot s2dv verification skill scores

Description

Plots *s2dv_cube* verification skill scores for gridded, polygon and point data generated by [c4h_verify](#). It is used for the following statistics: BSS, RPSS, CRPSS, AbsBiasSS, MSSS, RMSSS and ROCSS. For MSE and RMSE, please use the function [c4h_plot](#).

Usage

```

c4h_plotskill(
  skill,

```

```

    sign = NULL,
    time = NULL,
    boundaries = NULL,
    mask_boundaries = FALSE,
    title = NULL,
    legend = NULL,
    palette = "SunsetDark",
    noskillcol = "#b7d6ff",
    coordgrid = FALSE,
    ...
  )

```

Arguments

<code>skill</code>	<i>s2dv_cube</i> object with the skill data to be plotted.
<code>sign</code>	An additional optional binary <i>s2dv_cube</i> indicating significance.
<code>time</code>	Numerical index of the time to be plotted. Defaults to NULL (all times).
<code>boundaries</code>	Polygon boundaries to add to the map. Set to NULL for no boundaries (default), otherwise supply a polygon <i>sf</i> object or set to "countries" for coarse country boundaries.
<code>mask_boundaries</code>	If TRUE (default), mask the data to the provided boundaries.
<code>title</code>	Optional plot title.
<code>legend</code>	Optional legend title. If NULL (default), it uses the variable names.
<code>palette</code>	GHR, RColorBrewer or colorspace palette. Use "-" before the palette name (e.g., "-Reds") to reverse it. Alternatively, a vector of hex codes to create a continuous interpolated palette.
<code>noskillcol</code>	Colour for skill scores < 0 depicting a skill lower than climatologies.
<code>coordgrid</code>	If TRUE, display coordinate grids on the map.
<code>...</code>	Extra arguments indicating fixed aesthetics to be passed to <code>stars::geom_stars</code> , including <code>lwd</code> , <code>alpha</code> , <code>size</code> and <code>colour</code> .

Value

A *ggplot2* plot with the verification skill scores.

See Also

[c4h_verify](#) for calculating forecast skill and [c4h_plot](#) for general plotting of *s2dv* objects.

Examples

```

# DISCLAIMER: The purpose of these examples is to show the functionality
# of the different functions. To perform a meaningful verification, a much
# larger time series is required.
# Load data
hcst <- c4h_load(

```

```

    system.file("extdata", "hindcast", package = "clim4health"),
    variable = "t2m",
    year = 2010:2012,
    month = 1,
    leadtime_month = 1:3,
    ext = "nc"
  )
  lats <- hcst$coords$latitude
  lons <- hcst$coords$longitude
  obs <- c4h_load(
    system.file("extdata", "reanalysis", package = "clim4health"),
    variable = "t2m",
    year = 2010:2012,
    month = 1,
    leadtime_month = 1:3,
    bbox = c(max(lats), min(lons), min(lats), max(lons)),
    ext = "nc"
  )

# Example with gridded data
dwn <- c4h_downscale(
  "Interpolation",
  exp = hcst,
  obs = obs,
  method_remap = "bilinear",
  target_grid = system.file(
    "extdata",
    "reanalysis/t2m_201001.nc",
    package = "clim4health"
  )
)
hcst_dwn <- dwn$exp
skill_grid <- c4h_verify(exp = hcst_dwn, obs = obs, metrics = "BSS")
c4h_plotskill(skill_grid$BSS10$bss, skill_grid$BSS10$sign)
# Make some artificial significance for testing
skill_grid$BSS10$sign$data[,,, 3:8, 15:25] <- TRUE
c4h_plotskill(skill = skill_grid$BSS10$bss, sign = skill_grid$BSS10$sign)

# Example with polygon data
areas <- sf::read_sf(system.file("extdata", "areas", "munip_vallecauca.gpkg",
  package = "clim4health"))

obs_area <- c4h_space(obs,
  areas = areas,
  fun = "mean",
  areas_id = "munip_code")
hcst_area <- c4h_space(hcst,
  areas = areas,
  fun = "mean",
  areas_id = "munip_code")
skill_area <- c4h_verify(exp = hcst_area,
  obs = obs_area,
  metrics = "CRPSS",
  brier_thresholds = c(0.1, 0.9))

```

```

c4h_plotskill(skill_area$CRPSS$crpss, skill_area$CRPSS$sign)
# Make some artificial significance for testing
skill_area$CRPSS$sign$data[1, 1, 1, 1:3, 1, sample(1:42, 10)] <- TRUE
c4h_plotskill(skill = skill_area$CRPSS$crpss, sign = skill_area$CRPSS$sign)

# Example with station data
point_locs <- clim4health::c4h_load(system.file("extdata",
                                             "stations",
                                             package = "clim4health"),
                                  ext = "csv",
                                  year = 2010:2012,
                                  month = 1,
                                  leadtime_month = 1:3,
                                  variable = "temp_mean")
points <- list(latitude = point_locs$attrs$location$latitude,
              longitude = point_locs$attrs$location$longitude)
point_locs <- c4h_time(point_locs,
                      time_aggregation = "monthly",
                      dim_aggregation = "sdate",
                      fun = "mean")
dwn <- c4h_downscale("Interpolation", exp = hcst,
                    points = points, method_point_interp = "bilinear")
hcst_dwn <- dwn$exp
skill_point <- c4h_verify(exp = hcst_dwn,
                        obs = point_locs,
                        metrics = "BSS")
c4h_plotskill(skill = skill_point$BSS10$bss, boundaries = areas)
# Make some artificial significance for testing
skill_point$BSS10$sign$data[1, 1, 1, 1:3, 1, 4:6] <- TRUE
c4h_plotskill(skill = skill_point$BSS10$bss,
              sign = skill_point$BSS10$sign,
              boundaries = areas)

```

c4h_save

c4h_save

Description

Save *s2dv_cube* objects used in `clim4health` functions to disk.

Usage

```
c4h_save(data, path, drop = FALSE)
```

Arguments

`data` The *s2dv_cube* object to be saved.

path	A string indicating the path to save the file. The driver to save the file will be deduced based on the file extension. See details for a list of available extensions for each <i>s2dv_cube</i> data type.
drop	Logical. If TRUE, dimensions with only one value will be dropped when saving. If FALSE (default), they will be retained. Ignored when saving netCDF files.

Details

The list of available extensions is the following:

Gridded data 'nc', 'tif', 'csv'.

Areal data 'gpkg', 'shp'.

Point data 'csv'.

Note that when saving *s2dv_cube* objects an internal conversion using [c4h_convert](#) is applied.

Value

None.

Examples

```
fcst_path <- system.file("extdata", "forecast", package = "clim4health")
fcst <- c4h_load(fcst_path, variable = "t2m", ext = 'nc')
```

```
c4h_save(fcst, "out.nc")
file.remove("out.nc")
```

c4h_space	<i>Performs a spatial aggregation of gridded climate data into areas.</i>
-----------	---

Description

Performs a spatial aggregation of gridded climate data into areas.

Usage

```
c4h_space(data, areas, fun = "mean", weighting = "none", areas_id = NULL)
```

Arguments

data	<i>s2dv_cube</i> object with the climate data to be aggregates, usually read using c4h_load .
areas	Vector data, either a <i>sf</i> or <i>terra</i> object loaded into memory, or path to a vector file to be read. Contains the areal polygon geometries for the spatial aggregation.
fun	String. Method to perform the aggregation, defaults to "mean". Other possible options include "median", "sum", "min" and "max".

weighting	String that indicates whether weighted means should be used when fun = "mean" at the cost of increased computation. Options are "none" (a simple mean is applied to cells with centroid within the polygon), "approx" (for an approximate and fast weighting) and "exact" (for an exact weighting). See terra::extract for details.
areas_id	String. Name of the area identifier in the areas object.

Value

A s2dv_cube object with the aggregated data.

Examples

```
library(sf)

# Region municipalities (first 5)
municip_path <- system.file("extdata", "areas", "municip_vallecauca.gpkg",
                             package = "clim4health")
municip <- read_sf(municip_path)[1:5,]

# Forecast
fcst_path <- system.file("extdata", "forecast", package = "clim4health")
fcst <- c4h_load(fcst_path, variable = "t2m", ext = 'nc')

# Spatial aggregation
fcst_aggr <- c4h_space(fcst, municip)
```

c4h_time

c4h_time

Description

Performs a temporal aggregation of climate data from a finer to a coarser time scale.

Usage

```
c4h_time(
  data,
  time_aggregation,
  dim_aggregation,
  fun = "mean",
  week_start = NULL,
  min_valid = 1
)
```

Arguments

<code>data</code>	<code>s2dv_cube</code> object with the climate data to be aggregated, usually read using <code>c4h_load</code> .
<code>time_aggregation</code>	string indicating the type of temporal aggregations to be performed. Options are "monthly", "yearly", "weekly", and "daily".
<code>dim_aggregation</code>	string indicating the dimension along which the aggregation should be performed. Options are "sdate" and "time".
<code>fun</code>	string specifying the aggregation function. Options are "mean", "sum", "max", "min". Defaults to "mean".
<code>week_start</code>	for weekly aggregation, specify the starting weekday from "Monday" to "Sunday".
<code>min_valid</code>	integer specifying the minimum number of valid (non-NA) values desired in a group to return a non-NA aggregated value. Defaults to 1, so that any group with at least one valid value will be aggregated.

Value

An `s2dv_cube` object with the temporally aggregated climate data.

Examples

```
# Load station data
station_path <- system.file("extdata", "stations", package = "clim4health")
station_timeseries <- c4h_load(station_path, variable = "temp_mean",
                              ext = "csv")
stat_week <- c4h_time(station_timeseries, time_aggregation = "weekly",
                     dim_aggregation = "time", week_start = "Monday",
                     min_valid = 5)
dim(stat_week$data)

# Load station data with sdate dimension
station_data <- c4h_load(station_path, variable = "temp_mean", ext = "csv",
                        month = 1, leadtime_month = 1:12)
# Aggregate to monthly max across sdate dimension
stat_month <- c4h_time(station_data, time_aggregation = "monthly",
                       dim_aggregation = "sdate", fun = "max",
                       min_valid = 20)
dim(stat_month$data)
```

c4h_verify

c4h_verify

Description

Computes quality assessment metrics of climate forecasts.

Usage

```

c4h_verify(
  exp,
  obs,
  metrics,
  ref = NULL,
  brier_thresholds = c(0.1, 0.9),
  prob_thresholds = c(1/3, 2/3),
  alpha = 0.05,
  ncores = NULL,
  N.eff = NA,
  indices_for_clim = NULL,
  cross.val = FALSE,
  clim.cross.val = TRUE,
  weights_exp = NULL,
  weights_ref = NULL,
  comp_dim = NULL,
  limits = NULL,
  conf = TRUE,
  na.rm = FALSE,
  sign = TRUE,
  pval = FALSE,
  sig_method = NULL,
  sig_method.type = "two.sided.approx",
  rocss_cat = 1
)

```

Arguments

<code>exp</code>	An <code>s2dv_cube</code> object of forecast data with at least the 'sdate' and 'member' dimensions.
<code>obs</code>	An <code>s2dv_cube</code> object of observation data with at least the 'sdate' and 'member' dimensions.
<code>metrics</code>	A character vector of the verification metrics to be calculated. Valid options are: "BSS", "RPSS", "CRPSS", "AbsBiasSS", "MSE", "MSSS", "RMSE", "RMSSS", "ROCSS".
<code>ref</code>	An optional <code>s2dv_cube</code> object of reference data with at least the 'sdate' and 'ensemble' dimensions. The dimensions must be the same as 'exp' except 'ensemble'. If 'ref' is NULL, the climatological forecast is used as reference forecast. The default value is NULL.
<code>brier_thresholds</code>	Optional numeric vector of thresholds for the Brier Skill Score (BSS) calculation. The default is <code>c(0.1, 0.9)</code> . Only applicable if "BSS" is in <code>metrics</code> .
<code>prob_thresholds</code>	Optional numeric vector of thresholds for the Ranked Probability Skill Score (RPSS) calculation, and for the Relative Operating Characteristic Skill Score (ROCSS). The default is <code>c(1/3, 2/3)</code> . Only applicable if "RPSS" or "ROCSS" is in <code>metrics</code> .

alpha	A numeric of the significance level to be used in the statistical significance test. The default value is 0.05. Relevant for all metrics except "ROCSS". In some cases, a warning will be returned if alpha and sig_method.type are not compatible (e.g. if sig_method = "RandomWalk", sig_method.type is "two.sided.approx" and alpha is not equal to 0.05), and the significance level of the results will be indicated.
ncores	An integer indicating the number of cores to use for parallel computation. The default value is NULL.
N.eff	Effective sample size to be used in the statistical significance test. It can be NA (and it will be computed with the s2dv:::Eno), FALSE (and it will use the length of "obs" along "time", so the autocorrelation is not taken into account), a numeric (which is used for all cases), or an array with the same dimensions as "obs" except "time" (for a particular N.eff to be used for each case). The default value is NA. Only relevant if "BSS", "RPSS", "CRPSS", or "RMSSS" is in metrics.
indices_for_clim	A vector of the indices to be taken along 'sdate' for computing the thresholds between the probabilistic categories. If NULL, the whole period is used. The default value is NULL. Only relevant if "BSS", "RPSS", or "ROCSS" is in metrics.
cross.val	A logical indicating whether to compute the thresholds between probabilistic categories in cross-validation. The default value is FALSE. Only relevant if "BSS", "RPSS", or "ROCSS" is in metrics.
clim.cross.val	A logical indicating whether to build the climatological forecast in cross-validation (i.e. excluding the observed value of the time step when building the probabilistic distribution function for that particular time step). The default value is TRUE. Only used if 'ref' is NULL and "CRPSS" is in metrics.
weights_exp	A named numerical array of the forecast ensemble weights for probability calculation. The dimensions should include 'ensemble', 'sdate', and 'dataset' if there are multiple datasets. All dimension lengths must be equal to 'exp' dimension lengths. The default value is NULL, which means no weighting is applied. The ensemble should have at least 70 members or span at least 10 time steps and have more than 45 members if consistency between the weighted and unweighted methodologies is desired. Only relevant if "BSS" or "RPSS" is in metrics.
weights_ref	Same as 'weights_exp' but for the reference forecast.
comp_dim	A character string indicating the name of dimension along which obs is taken into account only if it is complete. The default value is NULL. Only relevant if "MSE" or "RMSE" is in metrics.
limits	A vector of two integers indicating the range along comp_dim to be completed. The default value is c(1, length(comp_dim dimension)). Only relevant if "MSE" or "RMSE" is in metrics.
conf	A logical value indicating whether to retrieve the confidence intervals or not. The default value is TRUE. Only relevant if "MSE" or "RMSE" is in metrics.
na.rm	A logical. FALSE means that NA values will not be removed, and the function will return 'NA' if they are present. TRUE means that NA values will

	be removed before computation. The default value is FALSE. Only relevant if "BSS", "RPSS", or "AbsBiasSS" is in <code>metrics</code> .
<code>sign</code>	A logical value indicating whether to compute or not the statistical significance of the test $H_0: (R)MSSS = 0$. The default value is TRUE. Only relevant if "MSSS" or "RMSSS" is in <code>metrics</code> .
<code>pval</code>	A logical value indicating whether to compute or not the p-value of the test $H_0: (R)MSSS = 0$. The default value is FALSE. Only relevant if "MSSS" or "RMSSS" is in <code>metrics</code> .
<code>sig_method</code>	A character string indicating the significance method. The options are NULL, "one-sided Fisher", "RandomWalk", and "DieboldMariano". Relevant for metrics "BSS", "RPSS", "CRPSS", "AbsBiasSS", "MSSS" and "RMSSS". The default is NULL. In this case, "RandomWalk" is applied for "BSS", "RPSS", "CRPSS", and "AbsBiasSS", and "one-sided Fisher" is applied for "MSSS" and "RMSSS".
<code>sig_method.type</code>	A character string indicating the test type of the significance method if <code>sig_method</code> is "RandomWalk" or "DieboldMariano". Valid options are "two.sided.approx", "two.sided", "greater", "less". The default is "two.sided.approx".
<code>rocss_cat</code>	An integer indicating the category to be returned when "ROCSS" is in <code>metrics</code> . The default value is 1, which means that the ROCSS for the first probability category (i.e. the lowest category) will be returned.

Value

a list of lists with the calculated metrics and their significances as `s2dv_cube` objects.

Note

See `clim4health_verification.Rmd` in the package vignettes for a detailed tutorial on function arguments and usage.

See Also

[c4h_plotskill](#) for visualizing the forecast skill.

Examples

```
# DISCLAIMER: The purpose of these examples is to show the functionality
# of the different functions. To perform a meaningful verification, a much
# larger time series is required.
# Load data
hcst <- c4h_load(
  system.file("extdata", "hindcast", package = "clim4health"),
  variable = "t2m",
  year = 2010:2012,
  month = 1,
  leadtime_month = 1:3,
  ext = "nc"
)
```

```

lats <- hcst$coords$latitude
lons <- hcst$coords$longitude
obs <- c4h_load(
  system.file("extdata", "reanalysis", package = "clim4health"),
  variable = "t2m",
  year = 2010:2012,
  month = 1,
  leadtime_month = 1:3,
  bbox = c(max(lats), min(lons), min(lats), max(lons)),
  ext = "nc"
)

# Example with gridded data
dwn <- c4h_downscale(
  "Interpolation",
  exp = hcst,
  obs = obs,
  method_remap = "bilinear",
  target_grid = system.file(
    "extdata",
    "reanalysis/t2m_201001.nc",
    package = "clim4health"
  )
)
hcst_dwn <- dwn$exp
skill_grid <- c4h_verify(exp = hcst_dwn, obs = obs, metrics = "BSS")
c4h_plotskill(skill_grid$BSS10$bss, skill_grid$BSS10$sign)

# Example with polygon data
areas <- sf::read_sf(system.file("extdata", "areas", "munip_vallecauca.gpkg",
  package = "clim4health"))
obs_area <- c4h_space(obs,
  areas = areas,
  fun = "mean",
  areas_id = "munip_code")
hcst_area <- c4h_space(hcst,
  areas = areas,
  fun = "mean",
  areas_id = "munip_code")
skill_area <- c4h_verify(exp = hcst_area,
  obs = obs_area,
  metrics = "CRPSS",
  brier_thresholds = c(0.1, 0.9))
c4h_plotskill(skill_area$CRPSS$crpss, skill_area$CRPSS$sign)

# Example with station data
point_locs <- clim4health::c4h_load(system.file("extdata",
  "stations",
  package = "clim4health"),
  ext = "csv",
  year = 2010:2012,
  month = 1,
  leadtime_month = 1:3,

```

```
                                variable = "temp_mean")
points <- list(latitude = point_locs$attrs$location$latitude,
              longitude = point_locs$attrs$location$longitude)
point_locs <- c4h_time(point_locs,
                      time_aggregation = "monthly",
                      dim_aggregation = "sdate",
                      fun = "mean")
dwn <- c4h_downscale("Interpolation", exp = hcst,
                    points = points, method_point_interp = "bilinear")
hcst_dwn <- dwn$exp
skill_point <- c4h_verify(exp = hcst_dwn,
                         obs = point_locs,
                         metrics = "BSS")
c4h_plotskill(skill = skill_point$BSS10$bss, boundaries = areas)
```

Index

c4h_collapse, [2](#)
c4h_convert, [4](#), [23](#)
c4h_convert_units, [5](#)
c4h_data, [6](#)
c4h_downscale, [7](#)
c4h_get, [10](#), [13](#)
c4h_get_help, [11](#), [12](#), [13](#)
c4h_index, [13](#)
c4h_load, [6](#), [7](#), [15](#), [23](#)
c4h_plot, [17](#), [19](#), [20](#)
c4h_plotskill, [18](#), [19](#), [28](#)
c4h_save, [22](#)
c4h_space, [23](#)
c4h_time, [24](#)
c4h_verify, [18–20](#), [25](#)