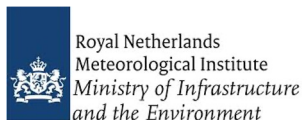


# icclim: a flexible tool to analyse climate simulations

Christian Pagé  
CERFACS, Toulouse, France



Abel Aoun

*CERFACS Toulouse, France*

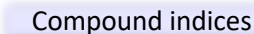
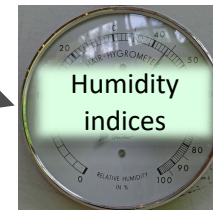
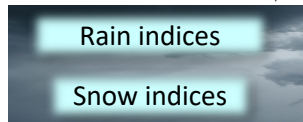
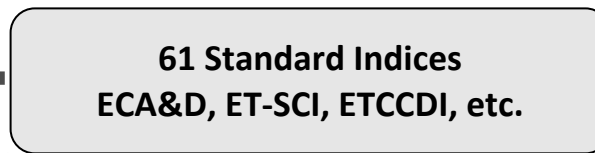
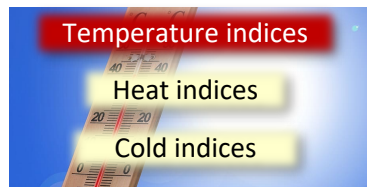
IS-ENES3 Webinar on the  
Climate4Impact portal and *icclim*  
April 13th, 2023 - 10:00-11:00 CEST



Max-Planck-Institut  
für Meteorologie



## icclim: Index Calculation for Climate



- Intra-period extreme temperature range [ $^{\circ}$  C] - **ETR**
- Warm days (days with mean temperature > 90th percentile of daily mean temperature) - **TG90p**
- Summer days (days with max temperature > 25  $^{\circ}$  C) - **SU**
- ...

- Python code developed at CERFACS since September 2013
- Funded by EC FP7 IS-ENES2, FP7 CLIPC and H2020 IS-ENES3
  - Generic and modular approach, can be reused in other environments
  - Starting from V5: completely rewritten and using underlying *xclim* functions, based on *xarray* and *dask*
  - I/O interface is structured for optimal performance
  - Implement the proper percentile indices calculations when calculation period overlaps reference period (called bootstrapping method)
  - Available indices: [https://icclim.readthedocs.io/en/latest/explanation/climate\\_indices.html#icclim-capabilities](https://icclim.readthedocs.io/en/latest/explanation/climate_indices.html#icclim-capabilities)



Documentation: [https://icclim.readthedocs.io/en/latest/python\\_api.html](https://icclim.readthedocs.io/en/latest/python_api.html)

Source code: <https://github.com/cerfacs-globc/icclim>

Current Version 6.2.0: <https://github.com/cerfacs-globc/icclim/releases/tag/v6.2.0>

## `icclim.index(**kwargs)`

- Parameters:**
- **in\_files** (*str* | *list[str]* | *Dataset* | *DataArray*) – Absolute path(s) to NetCDF dataset(s), including OPeNDAP URLs, or path to zarr store, or `xarray.Dataset` or `xarray.DataArray`.
  - **index\_name** (*str*) – Climate index name. For ECA&D index, case insensitive name used to lookup the index. For user index, it's the name of the output variable.
  - **var\_name** (*str* | *list[str]* | *None*) – **optional** Target variable name to process corresponding to `in_files`. If `None` (default) on ECA&D index, the variable is guessed based on the climate index wanted. Mandatory for a user index.
  - **slice\_mode** (*str*) – Type of temporal aggregation: {"year", "month", "DJF", "MAM", "JJA", "SON", "ONDJFM" or "AMJJAS"}. Default is "year". See `slice_mode` for details.
  - **time\_range** (*list[datetime.datetime]*) – **optional** Temporal range: upper and lower bounds for temporal subsetting. If `None`, whole period of input files will be processed. Default is `None`.
  - **out\_file** (*str* | *None*) – Output NetCDF file name (default: "icclim\_out.nc" in the current directory). Default is "icclim\_out.nc". If the input `in_files` is a `Dataset`, `out_file` field is ignored. Use the function returned value instead to retrieve the computed value. If `out_file` already exists, icclim will overwrite it!
  - **threshold** (*float* | *list[float]* | *None*) – **optional** User defined threshold for certain indices. Default depend on the index, see their individual definition. When a list of threshold is provided, the index will be computed for each thresholds.



<https://github.com/cerfacs-globc/icclim>

ci-build passing pypi v6.2.0 code style black docs passing conda-forge v6.2.0 Coverage 91% DOI 10.5281/zenodo.7643964

Fork 31 Starred 63 1,375 commits

```
import icclim
summer_days = icclim.su("netcdf_files/tasmax_1990-2100.nc", out_file="summer_days.nc")
```

- Findable:
  - [Open source code](#), licensed under the permissive Apache 2 license.
  - Try to follow [OpenSSF principles](#)
  - Documentation freely available on [readthedocs](#)
  - Dissemination in conferences (AGU, AMS, EGU...).
  - Integrated in [C4I](#) and C3S.
- Accessible
  - Documentation on readthedocs.
  - Github issues and pull requests.
  - CERFACS supports the development.
  - Can be installed with **pip** or **conda-forge**.
- Interoperable
  - Based on well knowns libraries: xarray, xclim, numpy.
  - Support NetCDF and xarray.Dataset input formats, as well as zarr.
  - Indices are deriviated from ECA&D and CF conventions standards.
  - icclim custom parsing for non CF complying inputs.
- Reusable
  - NetCDF output, with enriched metadata.
  - Easy to integrate within existing infrastructures.
  - Easy to write small scripts with it.

```
## KNMI TX
tx_files = glob.glob(f"netcdf_files/knmi/clean/*tx*.nc")
bp = [datetime.datetime(1901, 1, 1), datetime.datetime(1921, 12, 31)]
icclim.index(index_name='tx90p',
             in_files=tx_files,
             base_period_time_range=bp,
             slice_mode='YS',
             out_file="netcdf_files/output/out.nc")
```

## [new in 5.1.0] Rechunk how data is stored

Documentation: [https://icclim.readthedocs.io/en/latest/how\\_to/dask.html#create-an-optimized-chunking-on-disk](https://icclim.readthedocs.io/en/latest/how_to/dask.html#create-an-optimized-chunking-on-disk)

```
import icclim

ref_period = [datetime.datetime(1980, 1, 1), datetime.datetime(2009, 12, 31)]
with icclim.create_optimized_zarr_store(
    in_files="netcdf_files/tas.nc",
    var_names="tas",
    target_zarr_store_name="opti.zarr",
    keep_target_store=False,
    chunking={"time": -1, "lat": "auto", "lon": "auto"},
) as opti_tas:
    icclim.index(
        index_name="TG90p",
        in_files=opti_tas,
        slice_mode="YS",
        base_period_time_range=ref_period,
        out_file="netcdf_files/output/tg90p.nc",
    )
```

Documentation [https://icclim.readthedocs.io/en/latest/how\\_to/recipes\\_ecad.html#multi-index-computation](https://icclim.readthedocs.io/en/latest/how_to/recipes_ecad.html#multi-index-computation)

```
bp = [datetime.datetime(1991, 1, 1), datetime.datetime(1999, 12, 31)]
tr = [datetime.datetime(1991, 1, 1), datetime.datetime(2010, 12, 31)]
res = icclim.indices(
    index_group=IndexGroup.HEAT,
    in_files="./netcdf_files/climpact.sampledata.gridded.1991-2010.nc",
    base_period_time_range=bp,
    time_range=tr,
    out_file="pouetpouet.nc",
)
```



Documentation: [https://icclim.readthedocs.io/en/latest/references/ecad\\_functions\\_api.html](https://icclim.readthedocs.io/en/latest/references/ecad_functions_api.html)

```
import glob
import icclim
summer_days = icclim.su(
    in_files=glob.glob("netcdf_files/tasmax*.nc"),
    out_file="summer_days.nc",
)
```

## Functions

<code>tg</code> (in_files[, var_name, slice_mode, ...])	TG: Mean of daily mean temperature Source: ECA&D, Algorithm Theoretical Basis Document (ATBD) v11.
<code>tn</code> (in_files[, var_name, slice_mode, ...])	TN: Mean of daily minimum temperature Source: ECA&D, Algorithm Theoretical Basis Document (ATBD) v11.
<code>tx</code> (in_files[, var_name, slice_mode, ...])	TX: Mean of daily maximum temperature Source: ECA&D, Algorithm Theoretical Basis Document (ATBD) v11.
<code>dtr</code> (in_files[, var_name, slice_mode, ...])	DTR: Mean Diurnal Temperature Range Source: ECA&D, Algorithm Theoretical Basis Document (ATBD) v11.
<code>etr</code> (in_files[, var_name, slice_mode, ...])	ETR: Intra-period extreme temperature range Source: ECA&D, Algorithm Theoretical Basis Document (ATBD) v11.
<code>vdtr</code> (in_files[, var_name, slice_mode, ...])	vDTR: Mean day-to-day variation in Diurnal Temperature Range Source: ECA&D, Algorithm Theoretical Basis Document (ATBD) v11.

## [new in 5.3.0] custom season between exact dates



```
result = icclim.su(  
    in_files=xr.open_dataset(climp_file).tmax,  
    slice_mode=["season", ["19-07", "8 Aout"]],  
).compute()
```

# icclim v6

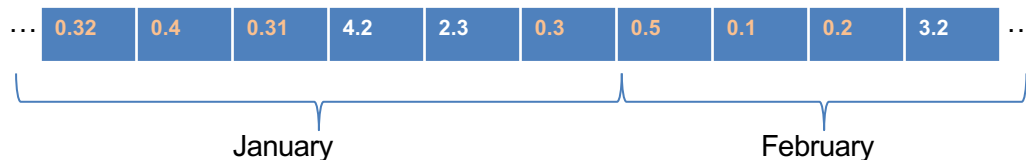
## Generic climate indices made easy



## [new in 6.0] Spells starting before the season bounds are properly counted



Precipitation



Say we want to compute cdd on a monthly basis...

CDD: number of consecutive where  $pr < 1$  mm/day

```
cdd = icclim.cdd(in_files=precipitation, slice_mode="month").CDD.compute()
```

Should February cdd be 3 days or 4 ?

With icclim 6 we count spells length before resampling into monthly values,  
So CDD in February would be 4 here

## [new in 6.0] Generic indices – What ?



What's the difference between SU, FD and Tx90p ?

Number of days where  $tx > 25 \text{ degC}$

Number of days where  $tn < 0 \text{ degC}$

Number of days where  $tx > 90\text{th (daily) percentile}$

An operator

A studied variable

A threshold

For our output, we want the right computation to be executed and to have specific metadata to clearly identify :

- What was the computation configuration
- What was the initial data used

## [new in 6.0] Generic indices – How ?



```
icclim.index(in_files=climp_file,  
             Operator   index_name="count_occurrences",  
             Studied variable var_name="tmax",  
             Threshold  threshold="≥ 20 deg_C",  
             ).compute()
```

**standard\_name:** number\_of\_days\_with\_maximum\_air\_temperature\_above\_threshold

**long\_name:** Number of days when maximum air temperature is greater or equal to 293.15 K for each year.

**units:** d

**cell\_methods:** time: sum over days



```
thresh = Threshold("≥ 75 period_per",  
                  threshold_min_value="1 mm/day",  
                  reference_period=['1991-01-01', "31 décembre 2000" ])  
r75p = icclim.index(climp_file,  
                   pr_thresh = Threshold("≥ 75 doy_per", threshold_min_value="1 mm/day")  
                   temp_thresh = Threshold("≥ 75 doy_per")  
                   c_w = icclim.index(climp_file,
```

R75p (ECAD)

CW (ECAD)

```
pr_thresh = Threshold("≥ 75 doy_per", threshold_min_value="1 mm/day")  
temp_thresh = Threshold("≥ 75 doy_per")  
wind_thresh = Threshold("≥ 95 doy_per")  
c_w_w = icclim.index(ds,  
                    index_name="count_occurrences",  
                    var_name=["precip", "tmax", "sfcWind"],  
                    threshold=[pr_thresh, temp_thresh, wind_thresh])
```

Cold, wet and windy  
(Bretagne?)

[https://icclim.readthedocs.io/en/latest/how\\_to/recipes\\_generic.html#generic-indices-recipes](https://icclim.readthedocs.io/en/latest/how_to/recipes_generic.html#generic-indices-recipes)

count_occurrences	SU, TR, TG90p, CD, R95p SD50cm ...
max_consecutive_occurrence	CSU, CFD, CDD, CWD
sum_of_spell_lengths	WSDI, CSDI
excess	GD4
deficit	HD17
fraction_of_total	R75pTOT, R95pTOT, r99pTOT
maximum	RX1day, TXx, TNx, custom__maximum
minimum	TXn, TNn, user_index - minimum
average	TG, TN, TX, SDII, SD, custom__mean
sum	PRCPTOT, custom__sum
standard_deviation	**new**
max_of_rolling_sum	RX5day, user_indexrolling_sum
min_of_rolling_sum	custom__rolling_sum
min_of_rolling_average	custom__rolling_mean
max_of_rolling_average	custom__rolling_mean
mean_of_difference	DTR
difference_of_extremes	ETR
mean_of_absolute_one_time_step_difference	vDTR
difference_of_means	custom_anomaly

## Average

Average of the `tas` variable, per year by default.

```
tas_average = icclim.average(in_files=data, var_name="tas").average.compute()
```

Average of the `tas` values that are above the 87th period percentile (computed on the whole period here), per year by default.

```
tas_average_above_percentile_of_period = icclim.average(
    in_files=data, var_name="tas", threshold="> 87 period_per"
).average.compute()
```

## Maximum Consecutive Occurrences

Almost equivalent to ECAD's index CDD (Consecutive Dry Days, days when pr is below 1 mm/day).

```
CDD = icclim.max_consecutive_occurrence(
    in_files=data, var_name="precip", threshold="< 1.3 mm/day"
).max_consecutive_occurrence.compute()
```

## Sum of Spell Lengths

Almost equivalent to ECAD's index WSDI (Warm Spell Duration Index, maximum consecutive occurrence of tasmax > 90th doy percentile)

```
custom_wsdi = icclim.sum_of_spell_lengths(
    in_files=data, var_name="precip", threshold="> 90 doy_per AND > 28 degC"
).sum_of_spell_lengths.compute()
```



<https://icclim.readthedocs.io/en/latest/references/threshold.html>



Scalar	Threshold(query=">=", value=25, unit="degC")
Day of year percentiles	Threshold(query=">=", value=99, unit="doy_per")
Period percentiles	Threshold(query=">=", value=99, unit="period_per")
Bounded period percentiles.	Threshold(">= 75 period_per". threshold_min_value="1 mm/day", reference_period=['1991-01-01', "31 décembre 2000" ])
Per-grid cell threshold	Threshold(query=">=", value="data.nc", threshold_var_name="tmin", unit="K")
Sequence of scalars	Threshold(query=">=", value=[25, 29], unit="degC")

## Examples

```
# -- Scalar threshold
scalar_t = build_threshold(">= 30 degC")
assert isinstance(scalar_t, BasicThreshold)

# -- Daily percentile threshold
doy_t = build_threshold(">= 30 doys_per")
assert isinstance(doy_t, PercentileThreshold)

# -- Per grid-cell threshold
grided_t = build_threshold(
    operator=">=", value="path/to/tasmax_thresholds.nc", unit="K"
)
assert isinstance(grided_t, BasicThreshold)

# -- Daily percentile threshold, from a file
tasmax = xarray.open_dataset("path/to/tasmax_thresholds.nc").tasmax
doys = xclim.core.calendar.percentile_doy(tasmax)
doy_file_t = build_threshold(operator=">=", value=doys)
assert isinstance(doy_file_t, PercentileThreshold)

# -- Bounded threshold
bounded_t = build_threshold(">= -20 degree AND <= 20 degree ")
# equivalent to:
x = build_threshold(">= -20 degree")
y = build_threshold("<= 20 degree")
bounded_t2 = x & y
assert bounded_t == bounded_t2
# equivalent to:
bounded_t3 = build_threshold(thresholds=[x, y], logical_link="AND")
assert bounded_t == bounded_t3
assert isinstance(bounded_t, BoundedThreshold)
```



IS-ENES Climate Data Infrastructure for Climate 4 Impact > C4I Use Cases as Jupyter Notebooks



## C4I Use Cases as Jupyter Notebooks

Project ID: 25761638

28 Commits 1 Branch 0 Tags 6.4 MB Project Storage

A collection of Jupyter Notebooks implementing some Use Cases.



### Update C4I\_userindex\_maxtemp\_freezing.ipynb

Christian Pagé authored 2 minutes ago

master notebooks / +

README Add LICENSE Add CHANGELOG Add CONTRIBUTING E out [9]:

Configure Integrations

Name	Last commit
<a href="#">C4I_Averaged_Temperature_Anomaly_2081...</a>	Updated with icclim 5.0.2.
<a href="#">C4I_Summer_days_Calculate_subset_and...</a>	Updated with icclim 5.0.2.
<a href="#">C4I_TX90p_Calculate_subset_and_plot.ip...</a>	Updated with icclim 5.0.2.
<a href="#">C4I_deltaT_deltaP_Anomaly_2081-2100_vs...</a>	Updated with icclim 5.0.2.
<a href="#">C4I_userindex_maxtemp_freezing.ipynb</a>	Update C4I_userindex_maxte
<a href="#">README.md</a>	Update README.md
<a href="#">cordex_download.ipynb</a>	Adjustments to dask config.

```
# Define plot
f, ax = plt.subplots(figsize=(14, 6),
                      subplot_kw={'projection': map_proj})

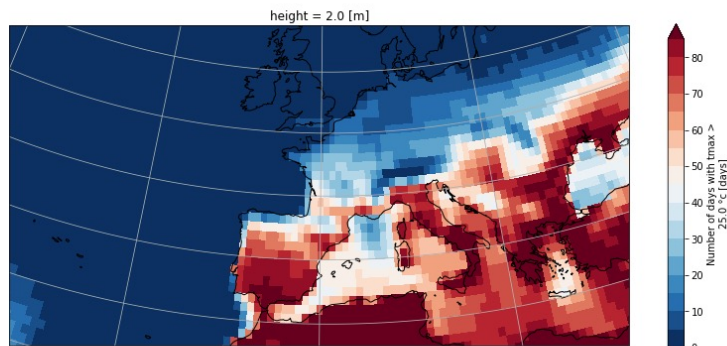
# Plot data with proper colormap scale range
levels = np.arange(0, 90, 5)
p = su_avg.plot(levels=levels,
                cmap='RdBu_r',
                transform=ccrs.PlateCarree())

# Plot information
plt.suptitle("Two Time Steps of Europe Summer Days", y=1)

# Add the coastlines to axis and set extent
ax.coastlines()
ax.gridlines()
ax.set_extent(extent)

# Save plot as png
plt.savefig('c4i_su_icclim.png')
```

Two Time Steps of Europe Summer Days



In [10]:

```
# Re-order longitude so that there is no blank line at 0 deg because 0 deg is within our spatial selection
```



# Thanks !



On behalf of the icclim team

<https://icclim.readthedocs.io/en/stable/>

Christian Pagé  
Abel Aoun


christian.page@cerfacs.fr

**THE CONSORTIUM**


Coordinated by CNRS-IPSL, the IS-ENES3 project gathers 22 partners in 11 countries





*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N°824084*




Our website  
<https://is.enes.org/>



Follow us on Twitter !  
**@ISENES\_RI**



Contact us at  
[is-enes@ipsl.fr](mailto:is-enes@ipsl.fr)



Follow our channel  
**IS-ENES3 H2020**

