

An Introduction

James Varndell



What is earthkit-plots?



earthkit-plots is the core **visualisation** component of **earthkit**.

earthkit-plots makes visualisation quick and easy with:

- a **concise**, **high-level** API
- **intelligent** formatting based on metadata
- **complex grids** supported out-of-the-box
- **customisable** style libraries

Why develop visualisation tools at all?

There are many fantastic Python libraries for data visualisation: **matplotlib, Cartopy, bokeh, Plotly, HoloViews, seaborn, Magics, metview,** and more.

earthkit-plots is **not** a new visualisation library - it is a **domain-specific** high-level library that **wraps lower-level tools** to drastically **reduce the boilerplate** needed for earth-science visualisation.

The goal is to let **scientists focus on their science**, not on the mechanics of plotting.

Software stack



matplotlib

earthkit geo

earthkit data

earthkit-plots uses **earthkit-data** to understand earth science datasets and **matplotlib** and **cartopy** to visualise them.

It has a very **high-level API** which hides complexity and greatly reduces boilerplate code.

Software stack





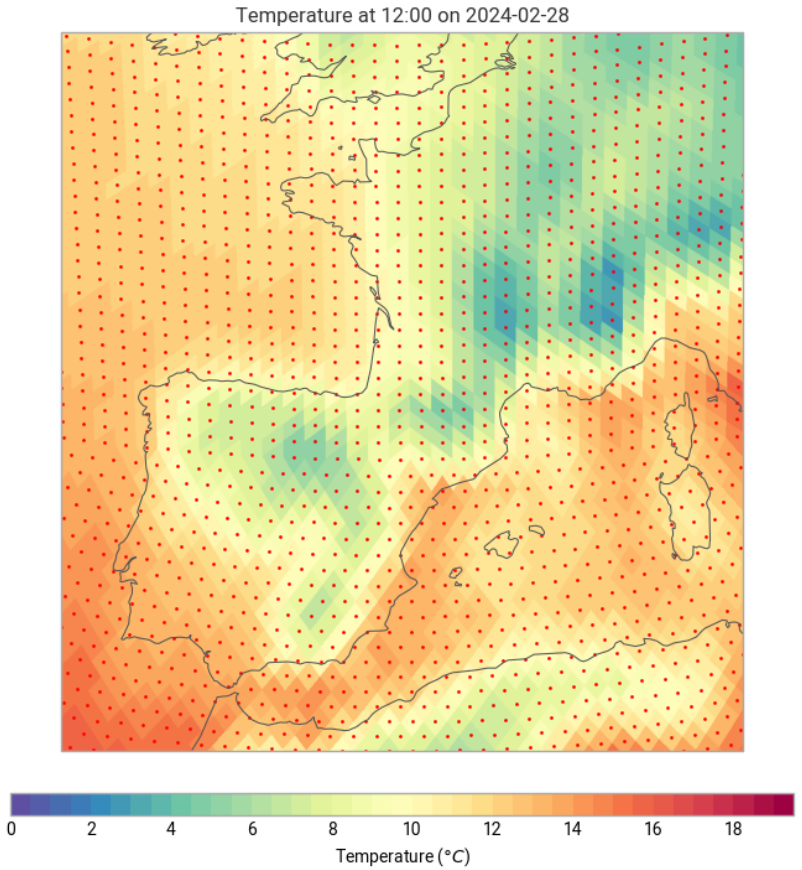




Building on top of widely adopted open-source tools allows us to **maintain less** and benefit from an existing, **well-documented API**.

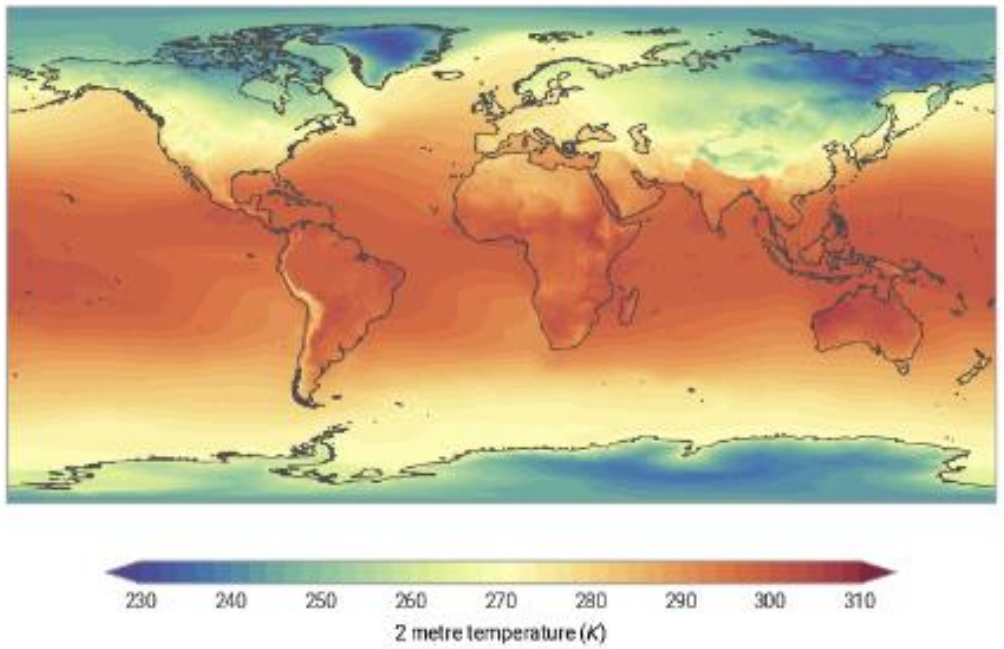
How does earthkit-plots make plotting easier?

- **High-level** functions do a lot with a little bit of code
- **Domain-specific knowledge** powers metadata formatting, unit conversion, automatic styles and more
- **Complex grids** (HEALPix, reduced Gaussian, ORCA etc.) are supported out-of-the-box
- Style libraries improve **consistency** of visualisations & branding



Ultra-high-level API

```
ekp.geo.plot(era5_2t)
```



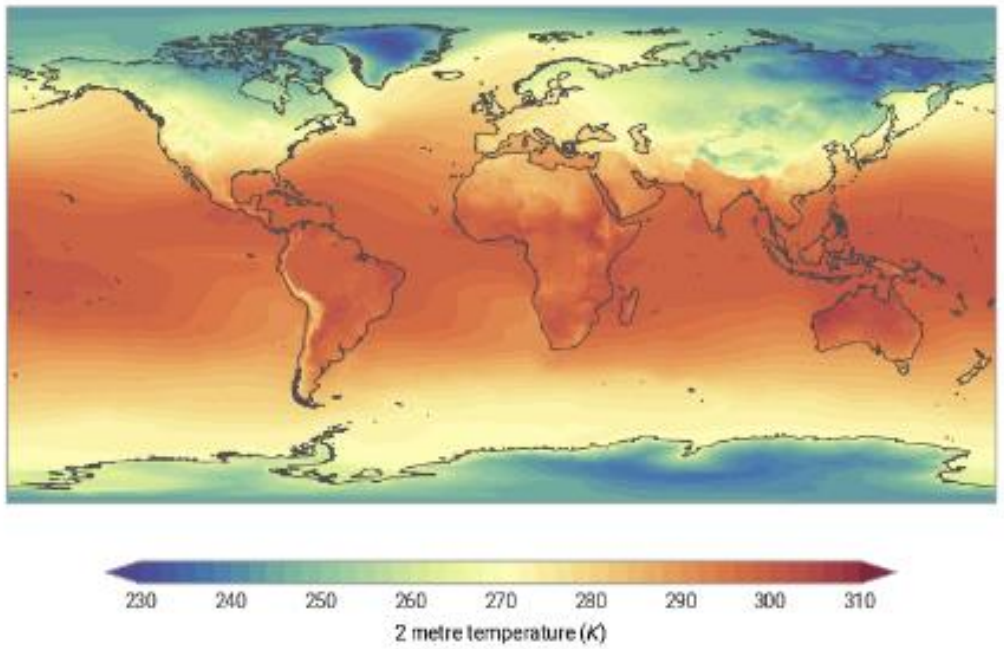
You can do a lot with a single line of code.

High-level API makes the “best attempt” at plotting data, including automatic styles.

That data can be **GRIB**, **netCDF**, **xarray** or **NumPy** arrays.

Ultra-high-level API

```
ekp.geo.plot(era5_2t)
```

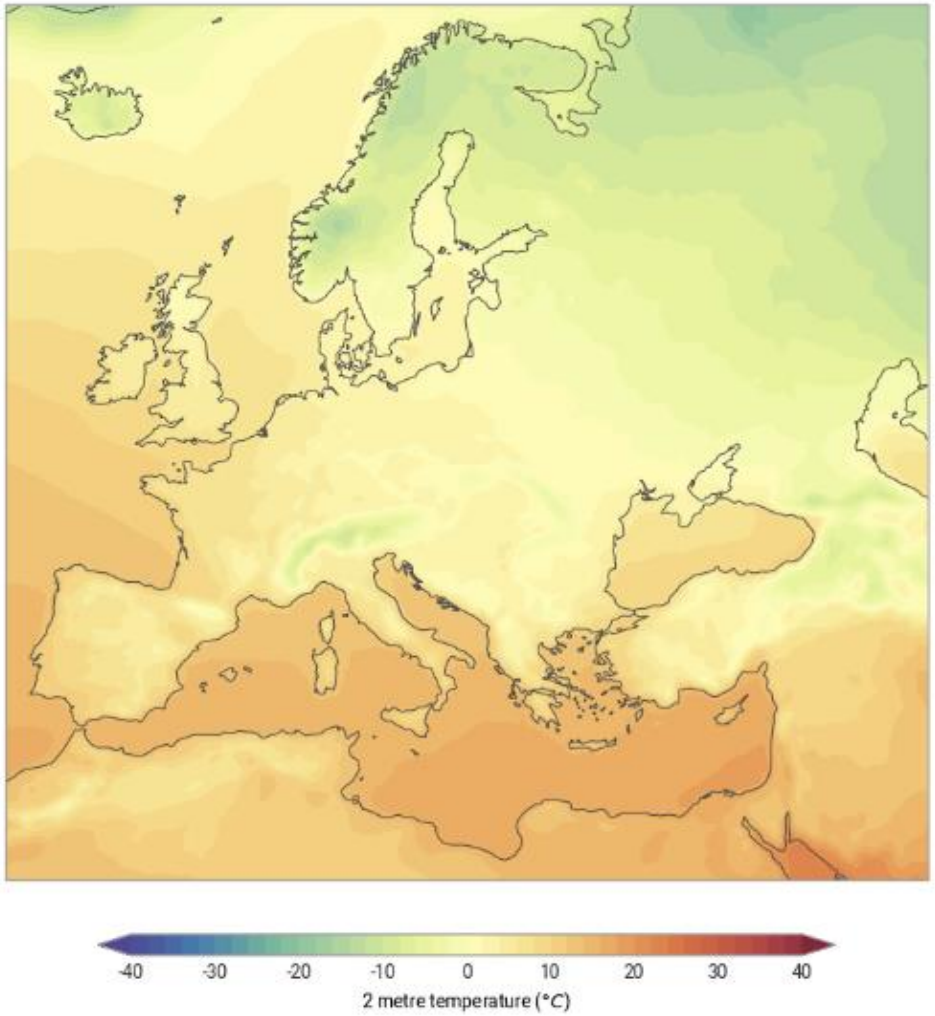


You can do a lot with a single line of code.

You do not need to extract coordinates or variables from your data: earthkit-plots does that automatically.

Domain and units

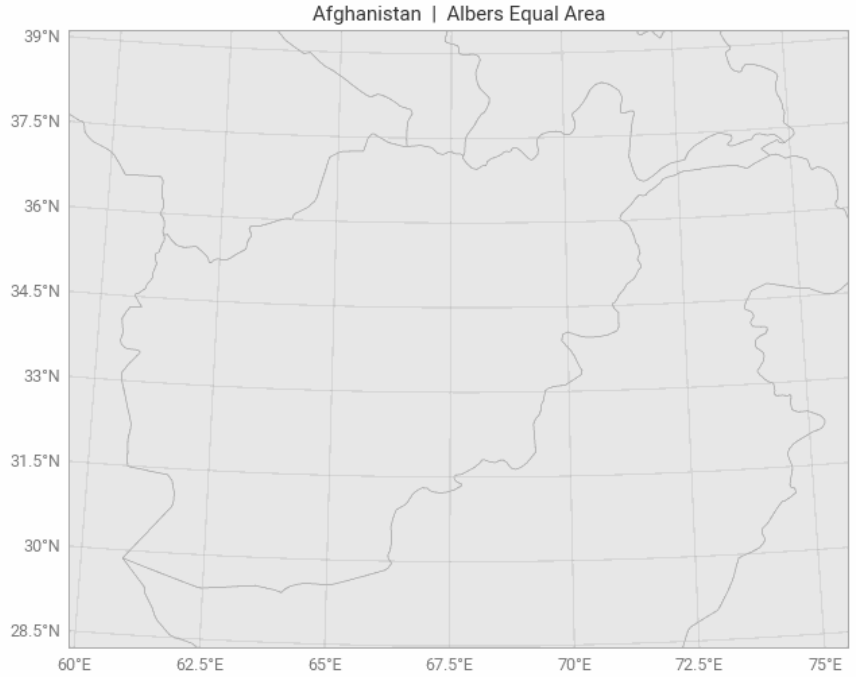
```
ekp.geo.plot(era5_2t, domain="Europe", units="celsius")
```



You can pass a **domain** and **units** to any plot to quickly tweak it to your needs.

By leveraging earthkit-data's understanding of data sources, we can do things like unit conversion on-the-fly.

Domains

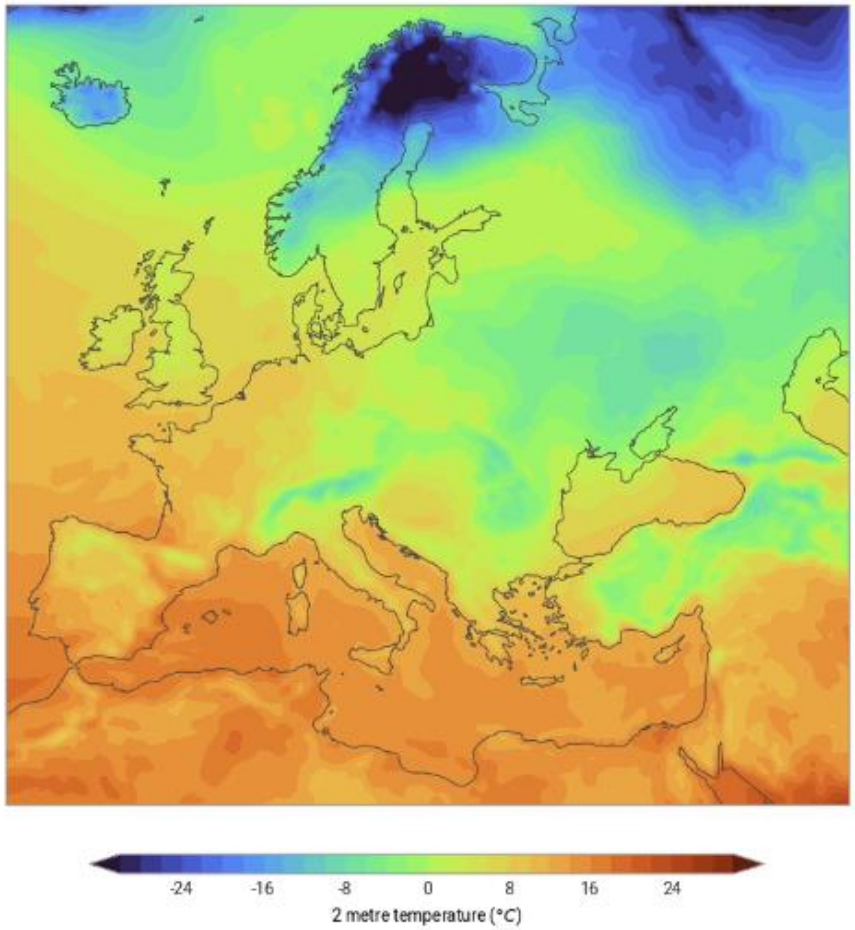


There are almost **200 built-in** domains (mostly named countries and continents) with **optimised map projections**.

You can also easily pass your own **custom domains**.

```
style = ekp.styles.Style(
    units="celsius",
    levels=range(-30, 31, 2),
    extend="both",
    cmap="turbo",
)

ekp.geo.plot(temperature, domain="Europe", style=style)
```



Styles

Styles define **how** your data is visualised.

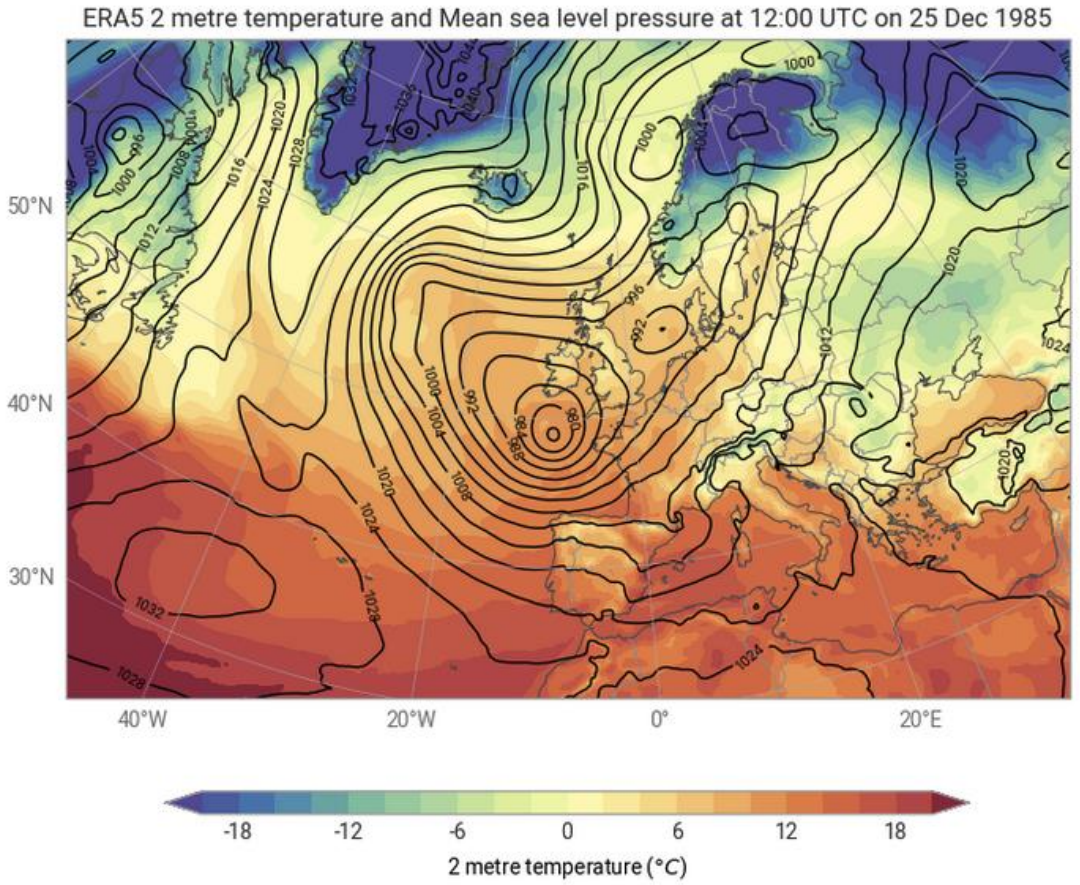
Mostly accept standard **matplotlib** arguments and can include **units**.

Easily **reusable** and reduce **cognitive load** of plotting methods…

…but plotting methods **still accept** standard **matplotlib** arguments.

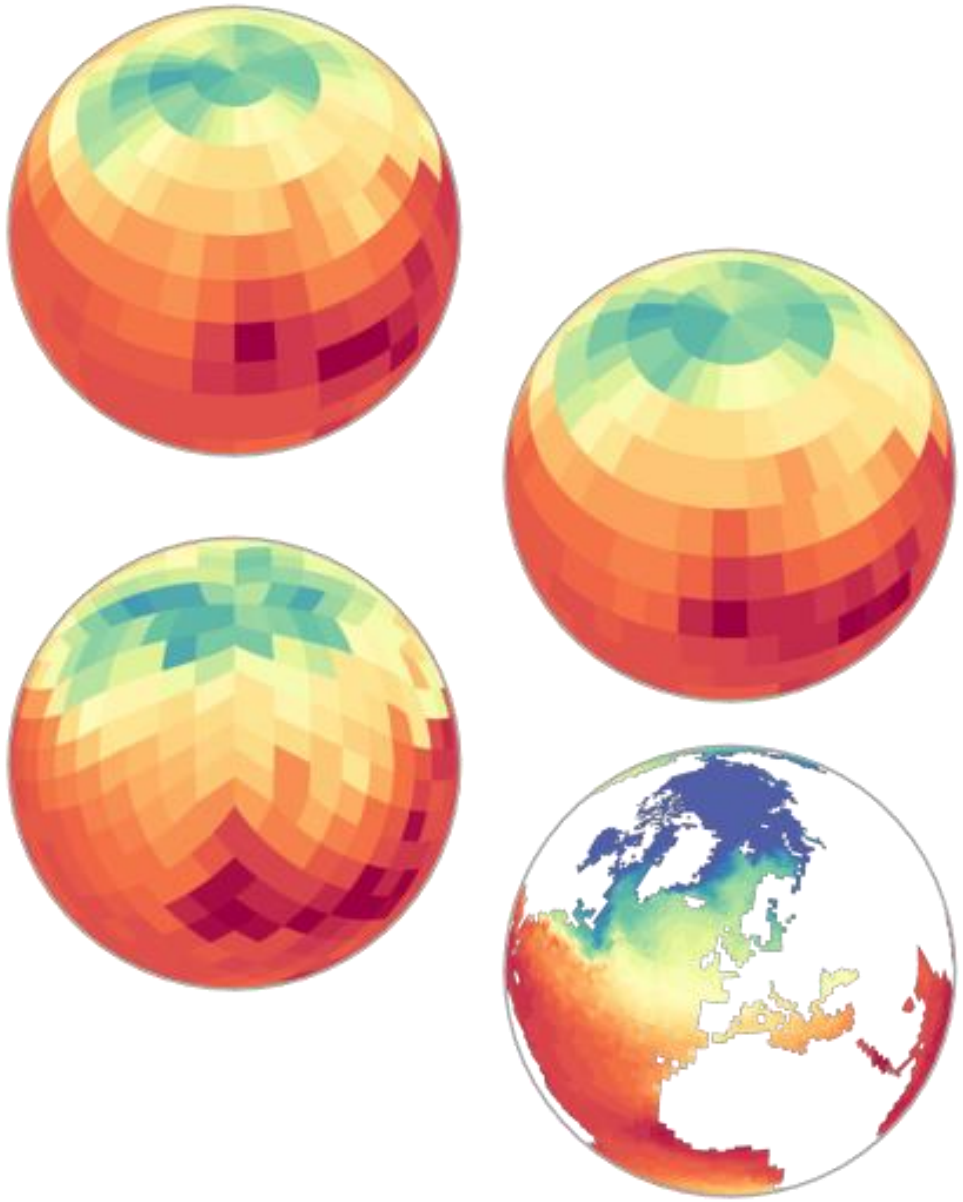
Metadata formatting

```
chart.title("ERA5 {variable_name} at {time:%H:%M UTC on %d %b %Y}")
```



Extract metadata directly in your titles, legend, axes labels and annotations.

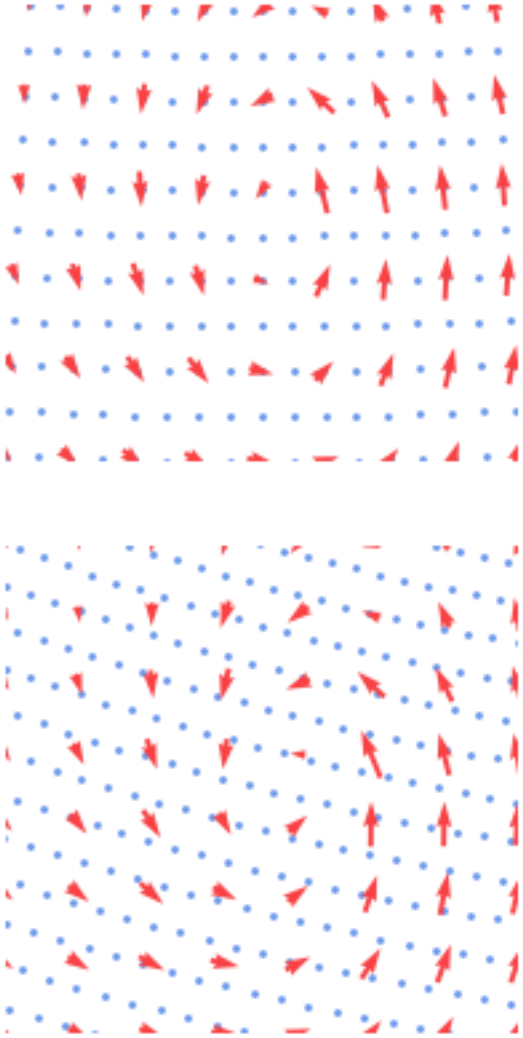
Makes it far easier to write **reusable** code which **adapts** to your data.



Grids

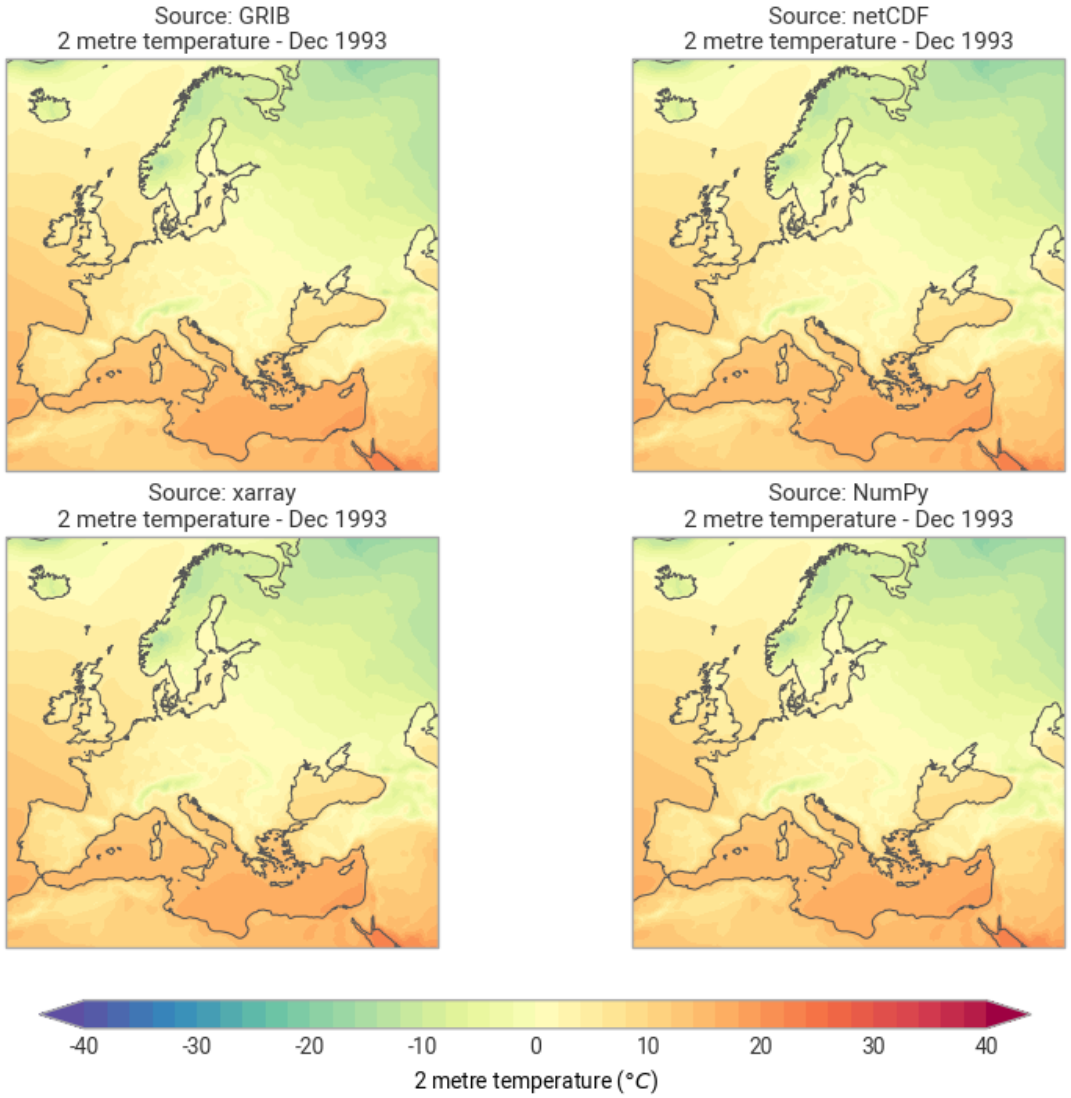
earthkit-plots is underpinned by earthkit-geo to make plotting **complex grids** (like Gaussian, HEALPix or ORCA grids) as easy as plotting regular latitude-longitude grids.

Resampling



A **high-level Pythonic API** for quickly subsampling or interpolating data before plotting

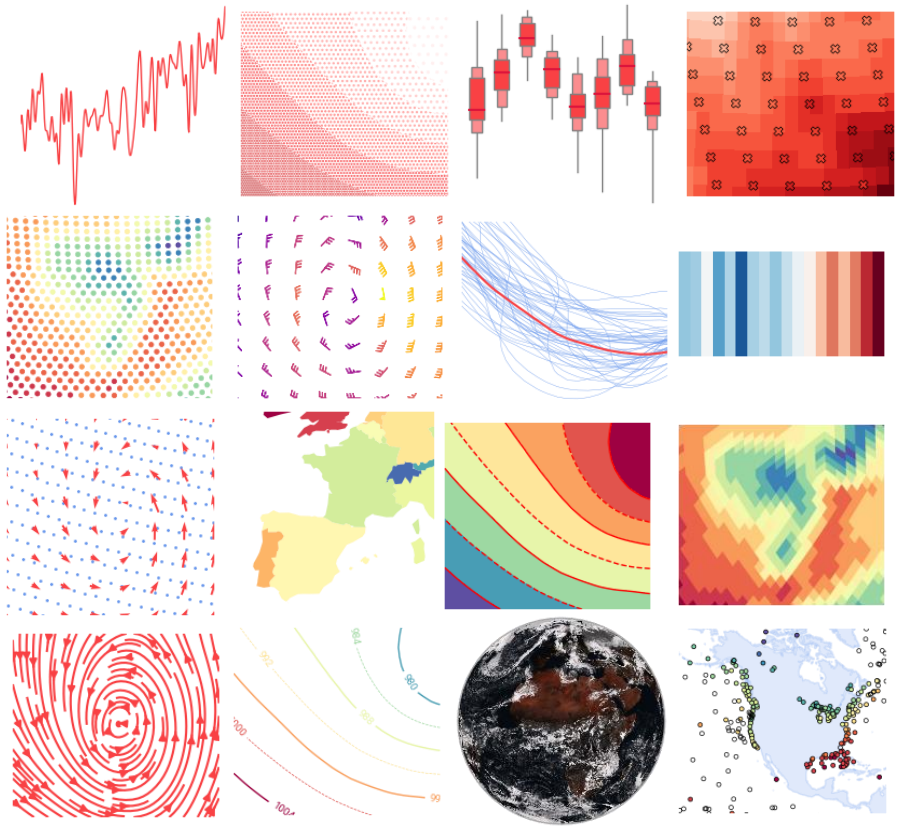
Source types



Since earthkit-data understands **many different sources** – GRIB, netCDF, xarray, NumPy and more – so does earthkit-plots!

Plotting each source uses **one consistent API**

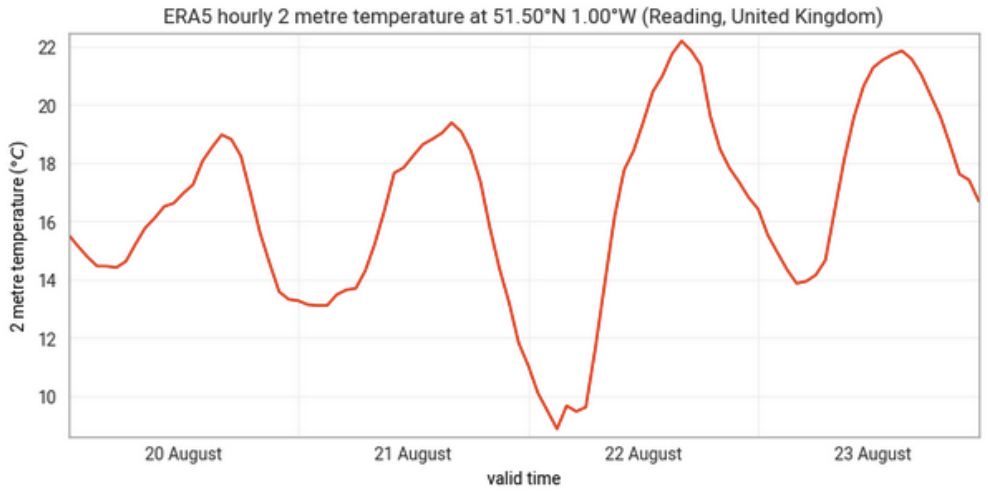
Plot types



Beyond standard meteorological plots, earthkit-plots has built-in methods for plotting **RGB composite** plots, **choropleth maps**, **meteograms**, **climate stripes** and more.

It’s not just maps…

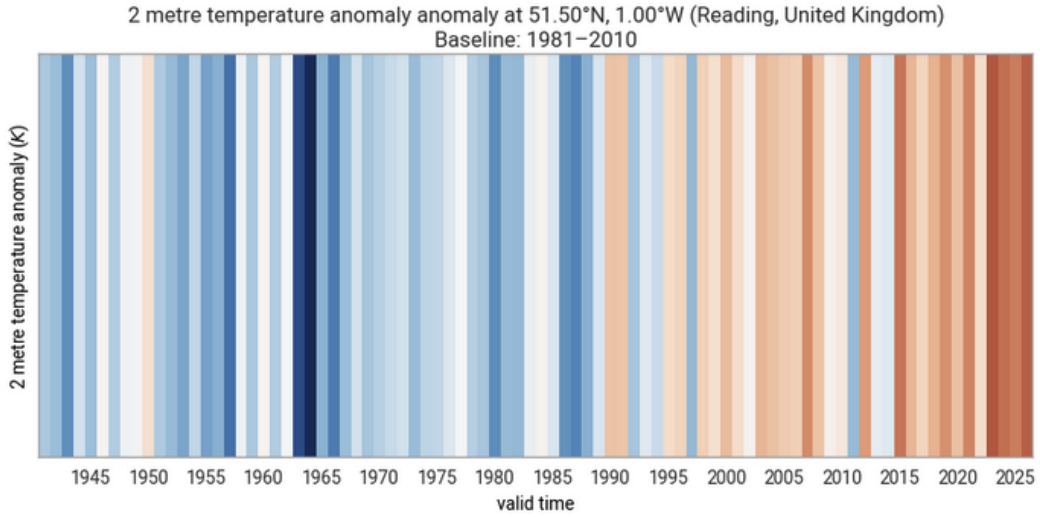
```python
ekp.timeseries.line(
    era5_reading_timeseries,
    color="red",
    # Metadata templating in the title, including location estimation
    title=(
        "ERA5 hourly {variable_name} at {latitude:%Lt} {longitude:%Ln}"
        "({location:%c}, {location:%C})"
    ).
    # High-level ticks formatting
    xticks={
        "frequency": "D",
        "format": "%d %B",
        "period": True,
    },
    # Unit conversion (source data is in Kelvin)
    units="celsius",
)
```



Features like **unit conversion**, **metadata templating**, and high-level APIs apply to other plot types too!

It’s not just maps…

```
(
    ekp.timeseries.stripes(anomaly, cmap="RdBu_r")
    .xticks(frequency="5Y")
    .title(
        "{variable_name} anomaly at {latitude:%Lt}, {longitude:%Ln} "
        "({location:%c}, {location:%C})\n"
        "Baseline: 1981–2010"
    )
    .show()
)
```



Features like **unit conversion**, **metadata templating**, and high-level APIs apply to other plot types too!

earthkit-plots provides high-level methods for many kinds of weather and climate visualisations

