The Verification Project

Claude Gibert

Language

- Use of a scripting language: Python.
  - Open Source.
  - Free.
  - Big community.
  - Lots of available tools and libraries, glues to open-source software.
- Time critical operations such as score computing are written in C++ and callable from Python.

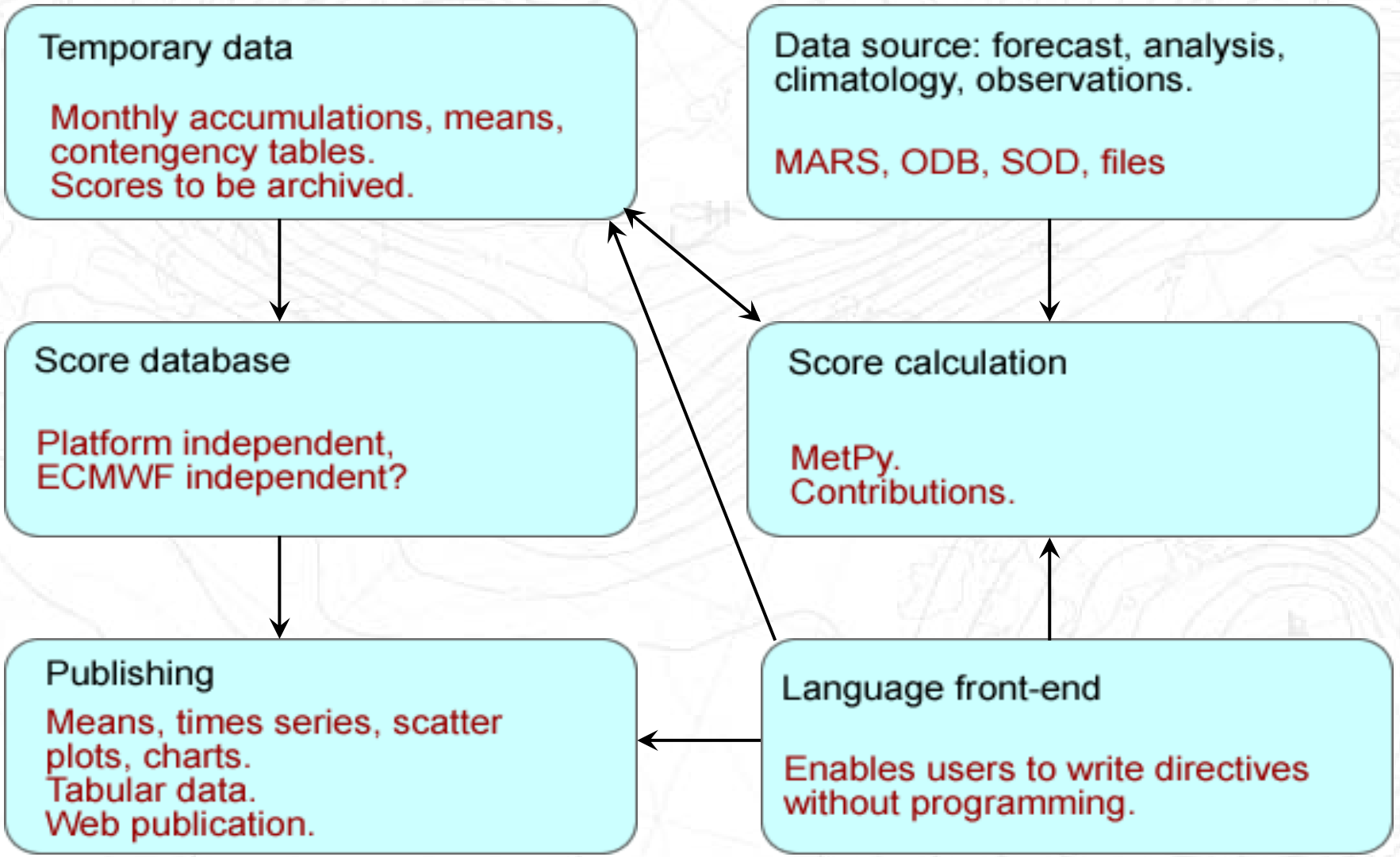
MetPy, current status

- The building blocks are in a library called MetPy:
  - GRIB decoding.
  - ODB access.
  - Score computing
  - Access to different file formats and databases (GRIB 1,2, NetCDF, Relational).
  - Plotting packages (Magics, open source).
  - Interface to MARS client, ECFS, Plot Content Management System etc…

Verify

- For the new verification system we will emphasise on:
  - Robustness.
  - Flexibility.
  - Openness, users have access to the source, they can also contribute.
  - Portability, some member state services have shown interest in using it on their systems.
  - Parameterisation, it should run on different systems, with different data formats.

Verification



Score calculation

- Current MetPy capabilities.
  - Deterministic, all the current scores:
    - Rmse, correlation coefficient, mean error, standard deviation in all possible combinations
  - Probabilitic scores:
    - Contingency tables, ROC area, cost-loss, reliability tables, Brier, Brier Skill scores.
- Scores against observations need to be included.

Data source

- MARS – FDB are the obvious sources of data. ODB will be used for easy access to data.
- In order for the new package to be flexible, files containing data in known format can be used to compute scores.
- An observation database could facilitate verification against observations.

Score Database

- The plan is to use netCDF files and see later if the format is suitable or needs to be changed.

Publishing

- Data publishing is obviously the goal of score computing:
  - Plots, means, time series, scatter plots, gains etc.. with their variations.
  - Table generation (ascii, excell?, binary).
  - Automatic web publication.

Front-end language

- A directive language is used to provide the end user with an easier interface to the package.
  Syntactically same style as the current one.
- In fact it is proper Python, but the user does not need to know it.
- This could simplify considerably verification scripts because MetPy provides support for date calculation, steps, etc…

Language

```
compute(
        param = Z,
        levtype = pl,
        levelist = (1000,500,100),
        score = (ancf,ref),
        steps = StepSequence(12,240,12),
        area = (‘europe’, ‘north hemisphere’),
        forecast = forecast (
        )
        persistence = persistence(
        )
        analysis = analysis (
                expver = ‘0001’,
                date = DateSequence(20040101,20040131),
        )
)
```