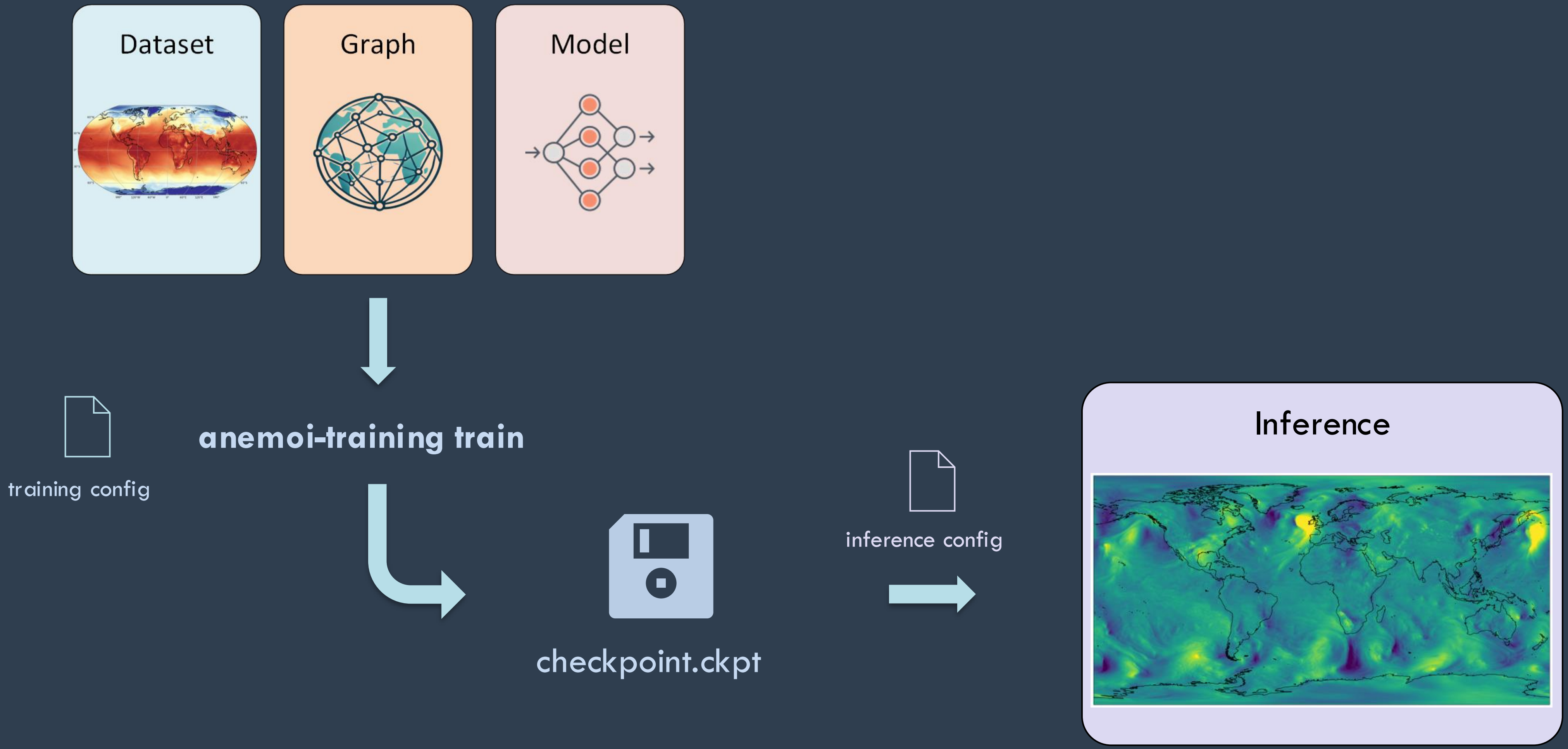
Anemoi Inference

Gert Mertes

Machine Learning Engineer

gert.mertes@ecmwf.int

On behalf of the Anemoi team and contributors



Dataset

Graph

Model

training config

anemoi-training train

checkpoint.ckpt

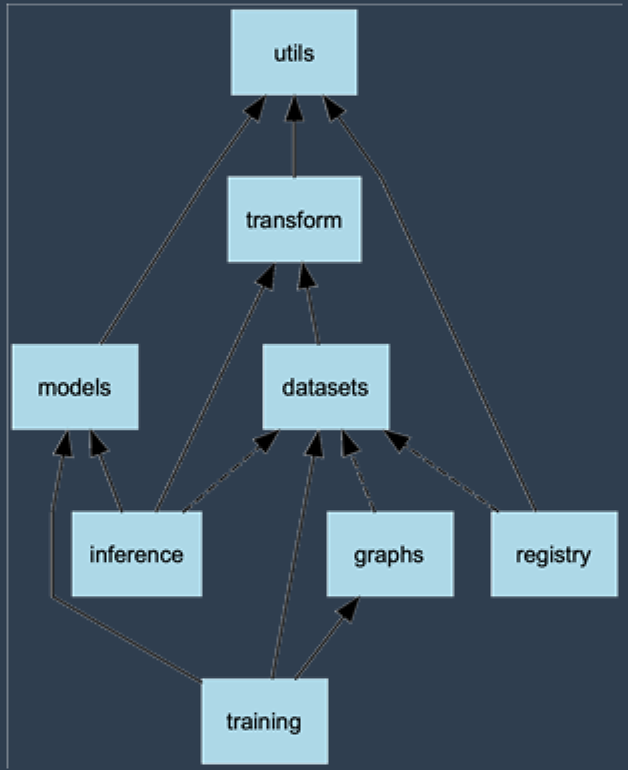
inference config

Inference

Why yet another package

- Run trained models in inference mode

- Community project, but built with operationalisation in mind
- Minimise unnecessary dependencies in operational suites
- Different data sources in training vs inference (zarr vs grib/netcdf/…)



Features

- Automatic discovery of model parameters from the checkpoint
  - Metadata embedded in the checkpoint
  - List of prognostic, diagnostic and forcing variables
  - Minimal to no configuration required to retrieve initial conditions

- User-extendable input and output
  - Multi-format I/O, e.g.: grib input, netcdf out
  - Inference input completely decoupled from training

- In progress: coupling of two or more models
  - E.g.: ocean / atmosphere

Runners

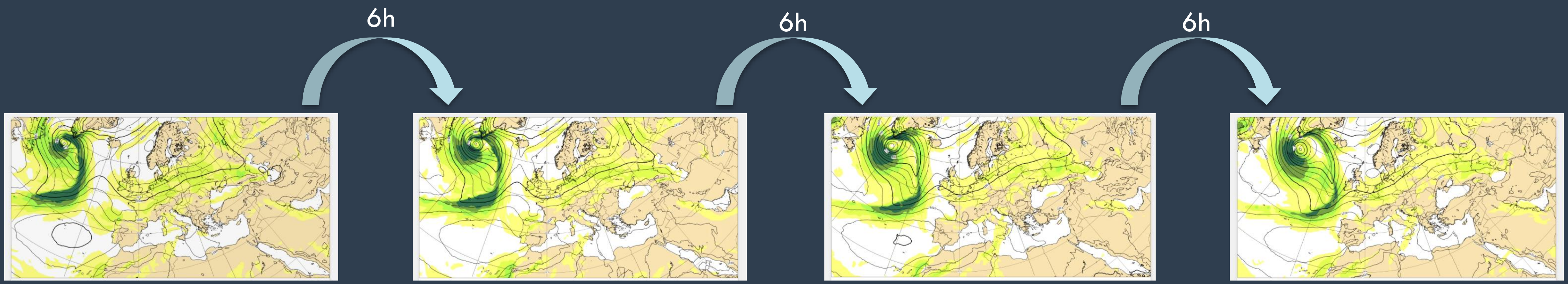
- Performs the inference loop (autoregressive rollout)
- Extendibility for special or exotic use-cases

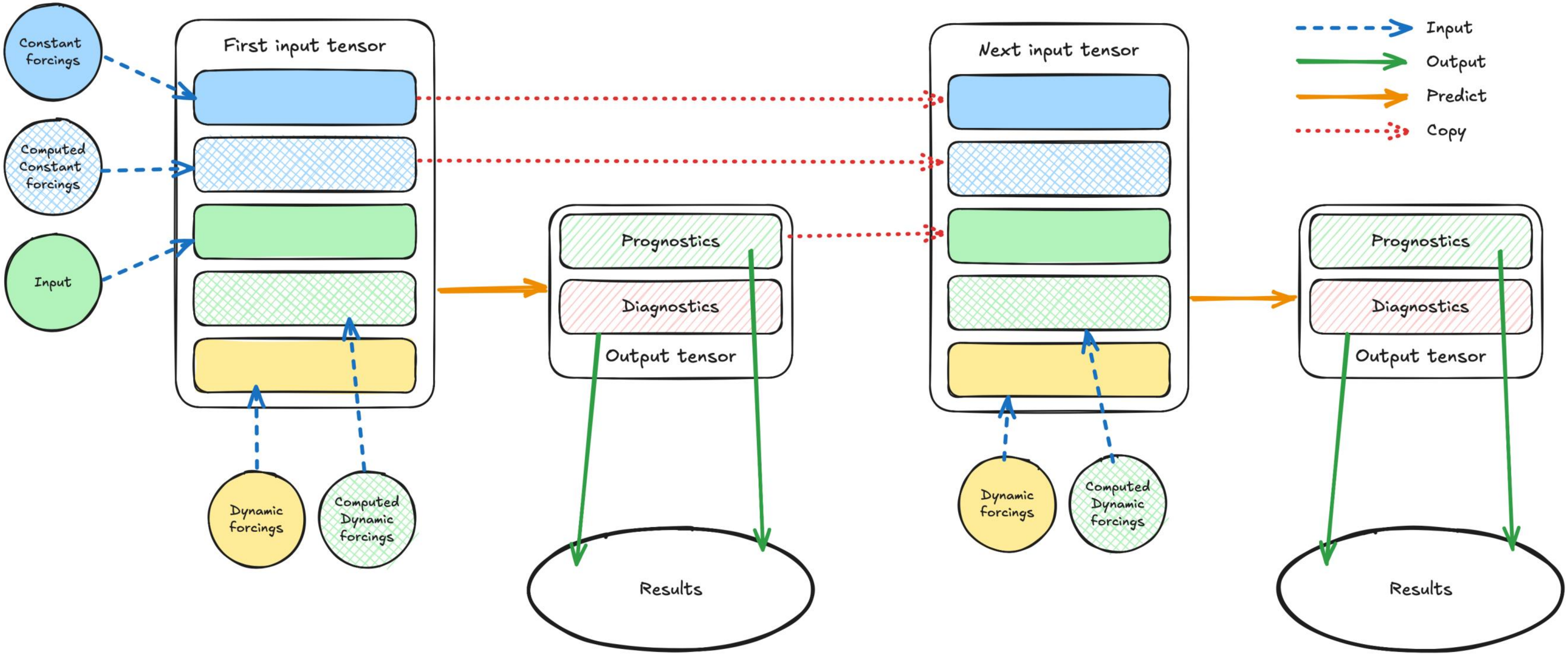
SimpleRunner

DefaultRunner

CutoutRunner

ParallelRunner




Constant forcings
Computed Constant forcings
Input
First input tensor
Dynamic forcings
Computed Dynamic forcings
Prognostics
Diagnostics
Output tensor
Results
Next input tensor
Dynamic forcings
Computed Dynamic forcings
Prognostics
Diagnostics
Output tensor
Results
Input
Output
Predict
Copy

Checkpoints and metadata

- Trained models are stored on disk as a checkpoint file
- Checkpoints contain metadata / lineage
  - Datasets used for training
  - Operations that took place on the data (e.g.: cutout)
  - Grid
  - Variables and their types (diagnostic, prognostic, forcing)
  - Mapping of variables to position in the input tensor
- Run inference without hard-coding anything

```
from anemoi.inference.checkpoint import Checkpoint

checkpoint = Checkpoint(‘checkpoint.ckpt')

checkpoint.variable_categories()
```

Three different API levels

- NumPy to NumPy low level API
- Object-oriented high level API
- Command-line API

https://anemoi-inference.readthedocs.io

https://github.com/ecmwf/anemoi-inference

work in progress

(welcome to contribute!)

NumPy to NumPy

```
import datetime
import numpy as np
from anemoi.inference.runners.simple import SimpleRunner
from anemoi.inference.outputs.printer import print_state

runner = SimpleRunner("checkpoint.ckpt")

latitudes = np.linspace(90, -90, 181) # 1 degree resolution
longitudes = np.linspace(0, 359, 360)

number_of_points = len(latitudes) * len(longitudes)
latitudes, longitudes = np.meshgrid(latitudes, longitudes)

input_state = {
        "date": datetime.datetime(2024, 10, 25),
        "latitudes": latitudes,
        "longitudes": longitudes,
        "fields": {
                "2t": np.random.rand(2, number_of_points),
                "msl": np.random.rand(2, number_of_points),
                "z_500": np.random.rand(2, number_of_points),
                ...: ...,
        },
}

for state in runner.run(input_state=input_state, lead_time=240):
        print_state(state)
```

Object oriented

```
import datetime

from anemoi.inference.inputs.gribfile import GribFileInput
from anemoi.inference.outputs.gribfile import GribFileOutput
from anemoi.inference.runners.default import DefaultRunner

runner = DefaultRunner("checkpoint.ckpt")

date = datetime.datetime(2024, 10, 25)

input = GribFileInput(runner, "input.grib")
output = GribFileOutput(runner, "output.grib")

input_state = input.create_input_state(date)

output.write_initial_state(input_state)

for state in runner.run(input_state=input_state, lead_time=240):
    output.write_state(state)

output.close()
```

Command-line API

```
$ anemoi-inference run –h

usage: anemoi-inference run [-h] [--defaults DEFAULTS] config [overrides ...]


Run inference from a config yaml file.


positional arguments:
  config               Path to config file.
  overrides            Overrides.


options:
  -h, --help           show this help message and exit
  --defaults DEFAULTS  Sources of default values.
```

Config format

```yaml
checkpoint: inference-checkpoint.ckpt
lead_time: 240
date: 2025-01-01T12:00:00
input:
  grib: input.grib
output:
  grib: output.grib
```

Inputs

- dataset
- grib
- mars
- cds

Outputs

- printer
- grib
- netcdf
- plot
- raw
- tee
- extract_lam

```
$ anemoi-inference run config.yaml

$ anemoi-inference run config.yaml date=2024-01-01T12:00:00
```

Demo time!

What’s next

- Coupling
  - Run two or more models at the same time
  - Coupled inputs and outputs
- Pre and post processing
- Suggestions or feature requests ?

Thank you!