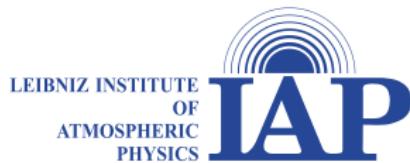


# MOD Meeting

Jared Frazier

16 July 2025



- ① OmniGlobe and OmniSuite
- ② omnisuite-viz
- ③ Miscellaneous: Better Acquisition of IFS Data

- ① OmniGlobe and OmniSuite
- ② omnisuite-viz
- ③ Miscellaneous: Better Acquisition of IFS Data

## OmniGlobe: What is it?

- *Spherical Display System:* OmniGlobe is a high-resolution spherical projection display that visually represents global data on a physical globe
- *Interactive and Customizable:* Supports dynamic and real-time content playback, **including user-created datasets and animations**, enabling interactive presentations and tailored visual storytelling.



Figure 1: OmniGlobe at KIT.

# OmniSuite: How do I preview "stories" for OmniGlobe?

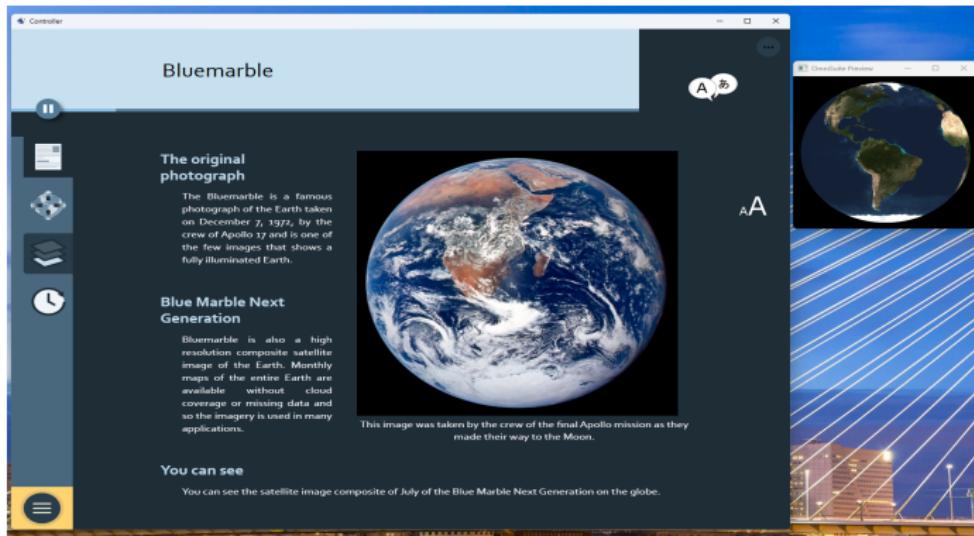


Figure 2: Example OmniSuite view of a stock "story". A story encapsulates assets (e.g., zonal wind animations) and metainformation (e.g., description) about the assets. Can also have multilayer assets (e.g., for vertical levels).

## Components of OmniSuite: What About Custom Stories?

- *Controller*: View "stories" (assets).
- *Catalog*: Download pre-made stories.
- **Material Editor**: Upload images to create the base images/animations for stories. Images can be created with Python package `omnisuite-viz`.
- **Story Editor**: Load in materials and add metainformation (descriptions, etc.).
- *Story Exporter*: Export pre-rendered stories.

- 1 OmniGlobe and OmniSuite
- 2 omnisuite-viz
- 3 Miscellaneous: Better Acquisition of IFS Data

## omnisuite-viz: Pypkg for Making OmniSuite Compatible Assets

- Provide generic interfaces for plotting arbitrary data on the globe.
- Ensure plots are compatible with format expected by OmniSuite.
- Produce animations of temporal geospatial data for diagnostic visualization independent of OmniSuite.
- [github/omnisuite-viz](https://github.com/omnisuite-viz)

## OmniSuite: Zonal Wind from Open Source Spectral Model

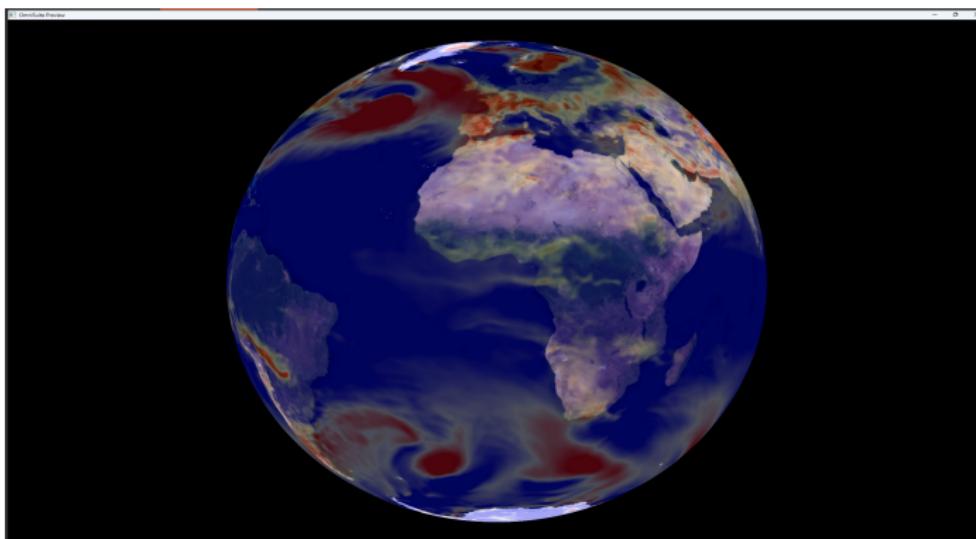


Figure 3: Preview of evolution of zonal wind data from 3D primitive equation model in SpeedyWeather.jl. Live demo after!

## Raw Animation: Zonal Wind from Open Source Spectral Model

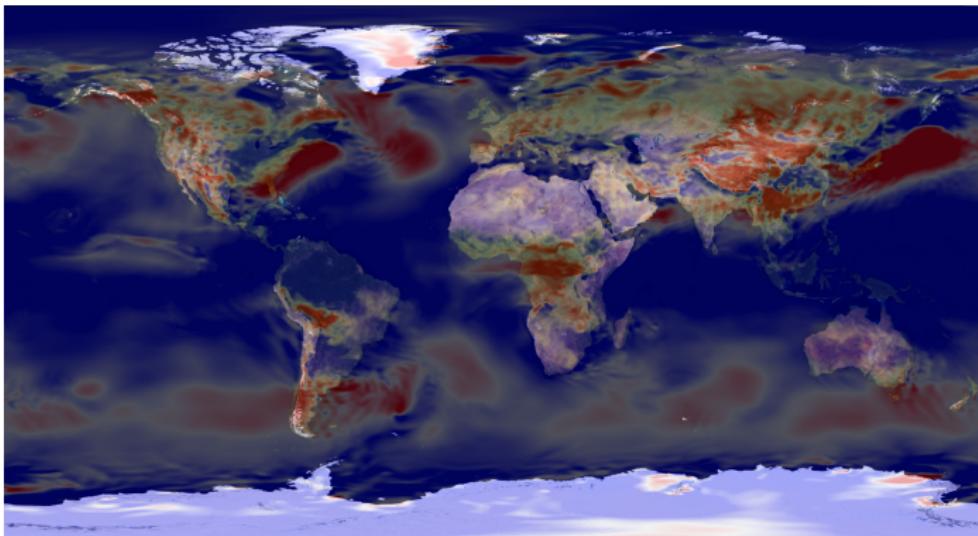


Figure 4: Raw asset for zonal wind data from 3D primitive equation model in SpeedyWeather.jl. Show animation after!

## OmniSuite: ICON Gravity Waves

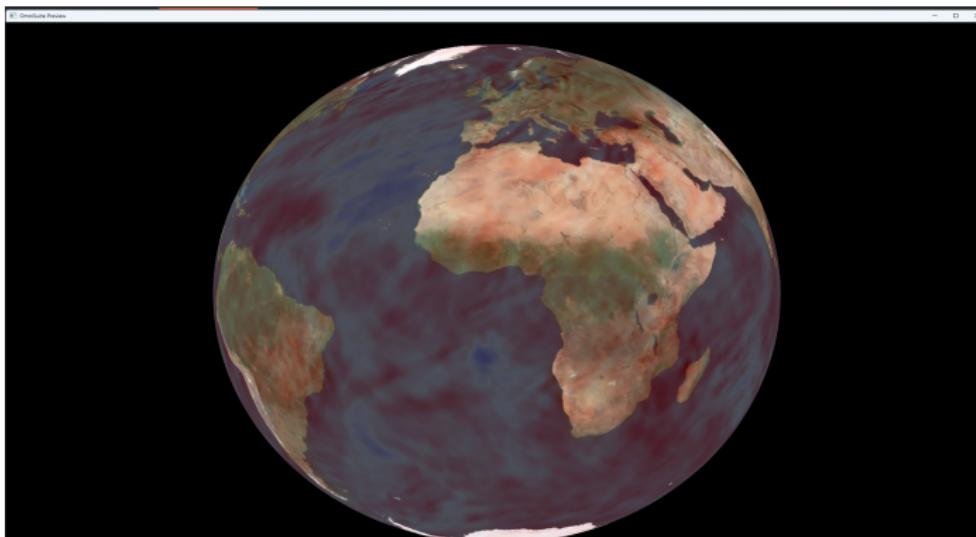


Figure 5: Preview of evolution of gravity waves from R2B7 ICON simulation. Live demo after!

## Raw Animation: ICON Gravity Waves

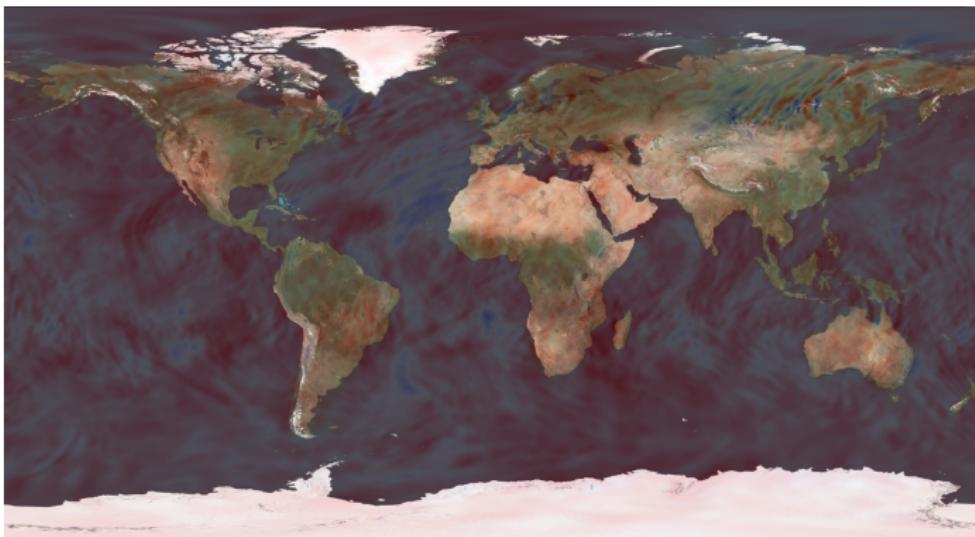


Figure 6: Raw asset for gravity wave data from ICON. Show animation after!

# Typical omnisuite-viz Workflow

- Subclass Reader to load NetCDF/Grib data and store it in a Grid object
- Subclass AnimatorConfig to control the asset output.
- Subclass Animator and override two methods.

```
77 # read netcdf data, the blue marble image, and post process
78 reader = IconDataReader(
79     netcdf_response_var_file_path=netcdf_response.var_file_path,
80     netcdf_file_name_of_response_var=netcdf.long_name_of_response_var,
81     netcdf_height_file_path=netcdf.height_file_path,
82     netcdf_long_name_of_height_var=netcdf.long_name_of_height_var,
83
84     blue_marble_path=blue_marble_path,
85
86     min_vertical_layer_height_in_meters=(
87         min_vertical_layer_height_in_meters),
88     max_vertical_layer_height_in_meters=(
89         max_vertical_layer_height_in_meters))
90
91 reader.read()
92 reader.postprocess()
93
94 grid = reader.grid
95 num_frames_in_animation = grid.response.shape[0] # equal to n timesteps
96
97 blue_marble_img = reader.blue_marble_img
98
99 # set up plotting configuration
100 config = NetcdfAnimatorConfig(
101     save_animation=save_animation,
102     output_dir=output_dir,
103
104     coastlines_kwarg={ 'lw': 0.0 },
105     plot_height_in_pixels=plot_height_in_pixels,
106     plot_width_in_pixels=plot_width_in_pixels,
107
108     netcdf_var_cmap_on_plot=cmap,
109     netcdf_var_transparency_on_plot=alpha,
110
111     num_frames_in_animation=num_frames_in_animation,
112
113     netcdf_response_var_file_path=netcdf_response.var_file_path,
114     blue_marble_path=blue_marble_path)
115
116 # write the frames to disk
117 animator = ICONModelAnimator(
118     grid=grid, config=config, blue_marble_img=blue_marble_img)
119
120 animator.animate()
```

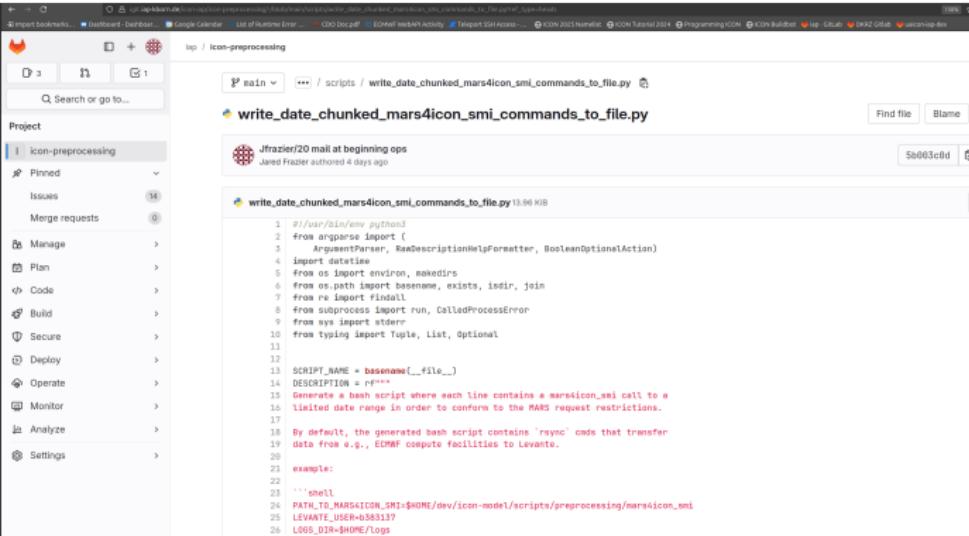
Figure 7: Example workflow to generate OmniSuite assets.

- ① OmniGlobe and OmniSuite
- ② omnisuite-viz
- ③ Miscellaneous: Better Acquisition of IFS Data

# Getting Operational/ERA-\* Data?

- *Problem:* Limited (2 TB) storage for login nodes on ECMWF's HPC system called ATOS.
- *Goal:* Efficiently transfer Operational, ERA-5, ERA-Interim, or ERA-40 data from these login nodes to Levante.
- *Solution:* Automated transfer and deletion of IFS data suitable for ICON simulations to Levante. Prior solution only acquired a more limited form of operational data.
- *Limitation:* Still requires ECMWF credentials—no workarounds to this.

# Automated Solution



The screenshot shows a GitHub repository interface for the project "icon-preprocessing". The repository contains a file named "write\_date\_chunked\_mars4icon\_smi\_commands\_to\_file.py". The code in the file is as follows:

```
1 #!/usr/bin/env python3
2 from argparse import (
3     ArgumentParser, RawDescriptionHelpFormatter, BooleanOptionalAction)
4 import datetime
5 from os import environ, makedirs
6 from os.path import basename, exists, isdir, join
7 from re import findall
8 from subprocess import run, CalledProcessError
9 from sys import stderr
10 from typing import Tuple, List, Optional
11
12
13 SCRIPT_NAME = basename(__file__)
14 DESCRIPTION = r'''**
15 Generate a bash script where each line contains a mars4icon_smi call to a
16 limited date range in order to conform to the MARS request restrictions.
17
18 By default, the generated bash script contains 'rsync' cmds that transfer
19 data from e.g., ECMWF compute facilities to Levante.
20
21 example:
22
23 '''shell
24 PATH_TO_MARS4ICON_SMI=$HOME/dev/icon-model/scripts/preprocessing/mars4icon_smi
25 LEVANTE_USER=d383137
26 LOGS_DIR=$HOME/logs
```

Figure 8: Script automating ERA-\* data transfer to Levante. See [icon-preprocessing/scripts/](#). For reference, c.a. 5 days to transfer 20 years of ERA-5 data (c.a. 20 TB).