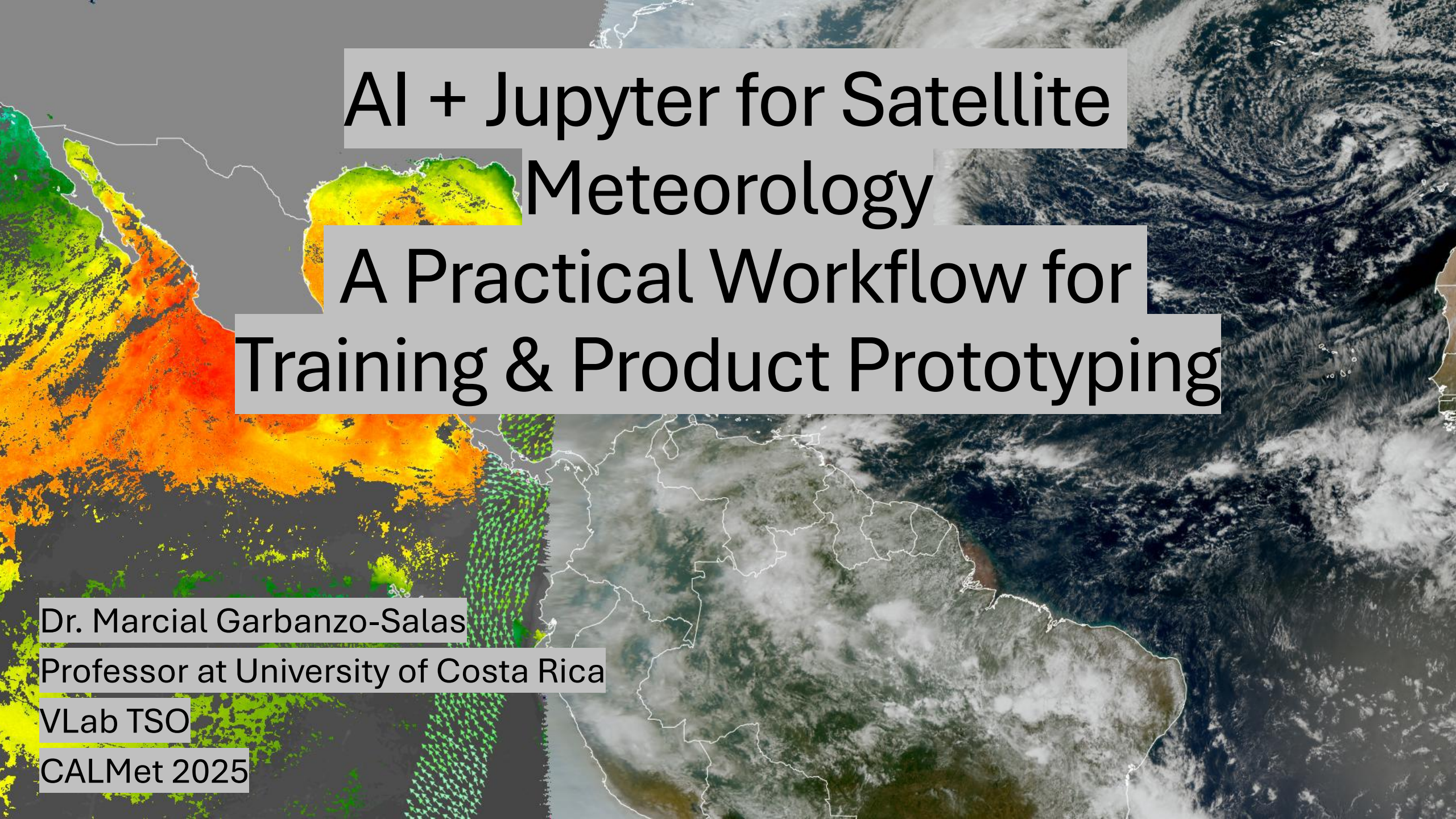
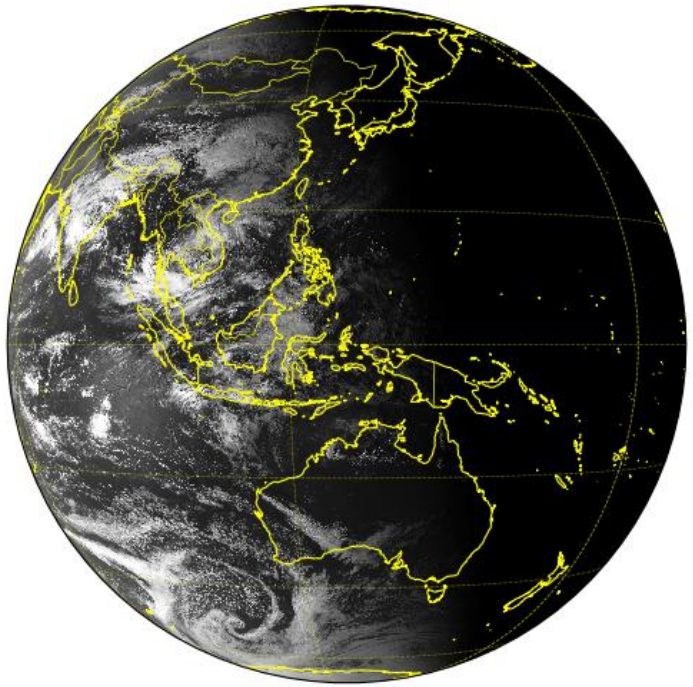


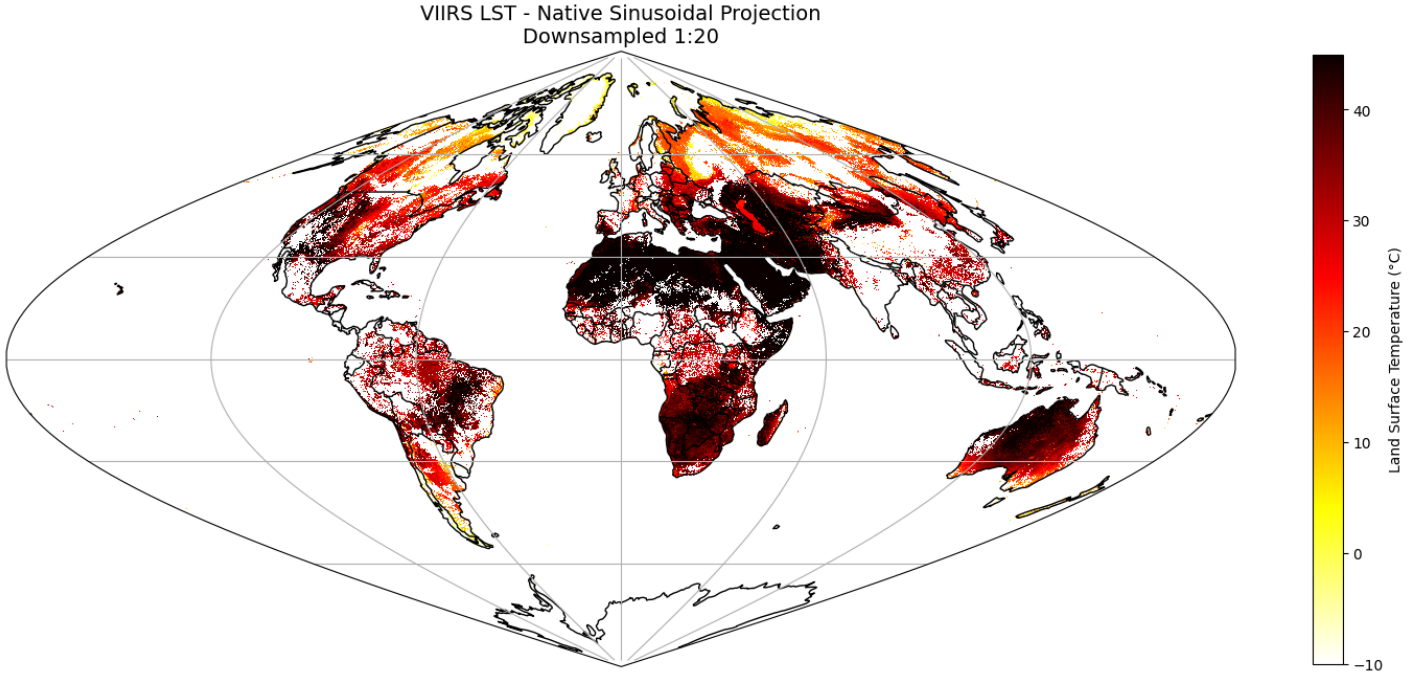
AI + Jupyter for Satellite Meteorology
A Practical Workflow for Training & Product Prototyping

Dr. Marcial Garbanzo-Salas
Professor at University of Costa Rica
VLab TSO
CALMet 2025

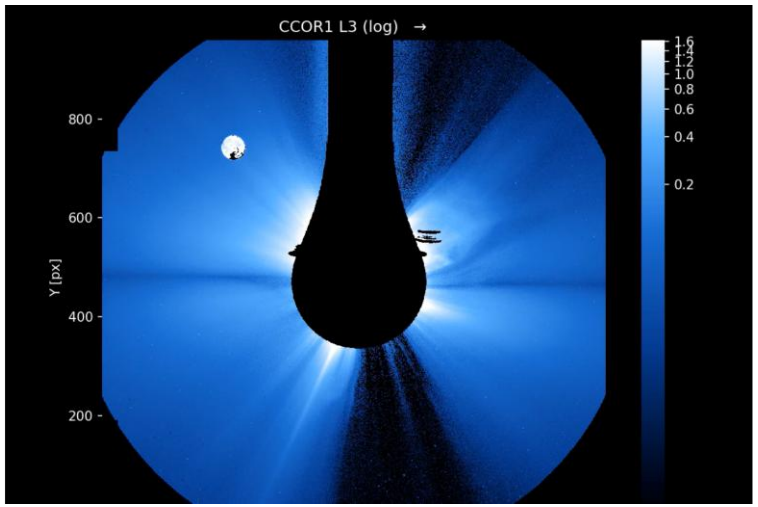
Why AI? Why Now?

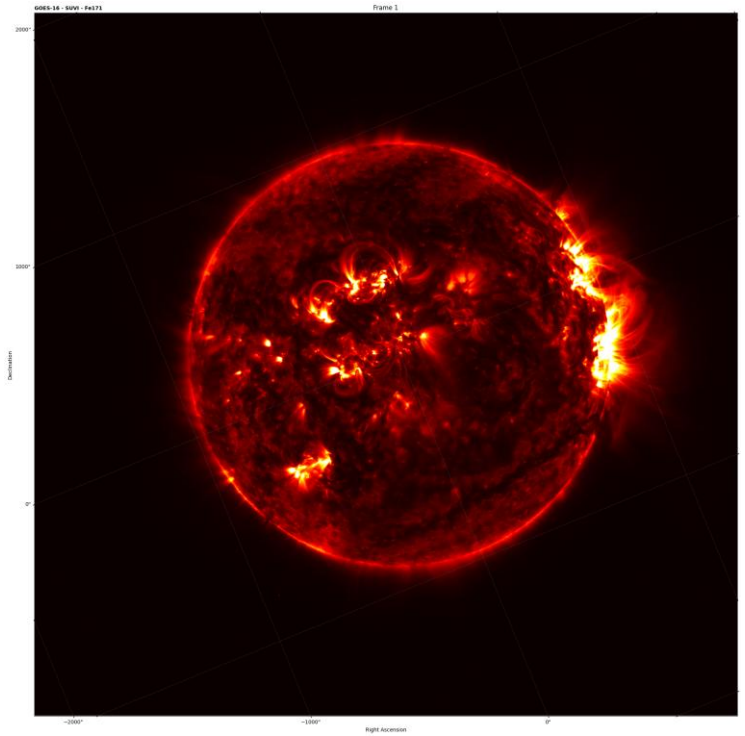
- AI accelerates coding, visualization, and teaching.
- Satellite datasets are becoming richer (ABI, MTG, GLM, etc.).
- Trainers and developers need faster prototyping workflows.

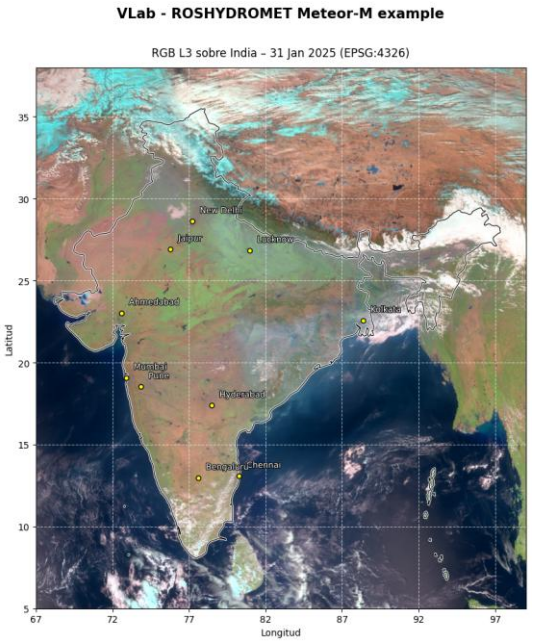




Because it works!







What AI *cannot* do (yet)

Meaning

- It does **not** understand (some) physical meaning.

Errors

- It produces **errors and hallucinations** (sometimes).

Maintenance

- Code is often **not maintainable** without refactoring (if not created correctly)

Mentoring

- We must **guide it** with good prompts (always).

What AI *can* do

Generate **initial code** for download, analysis or visualization

Refactor messy code into clean, reusable functions

Produce **documentation**, comments, and explanations

Suggest **workflows**, steps, and best practices

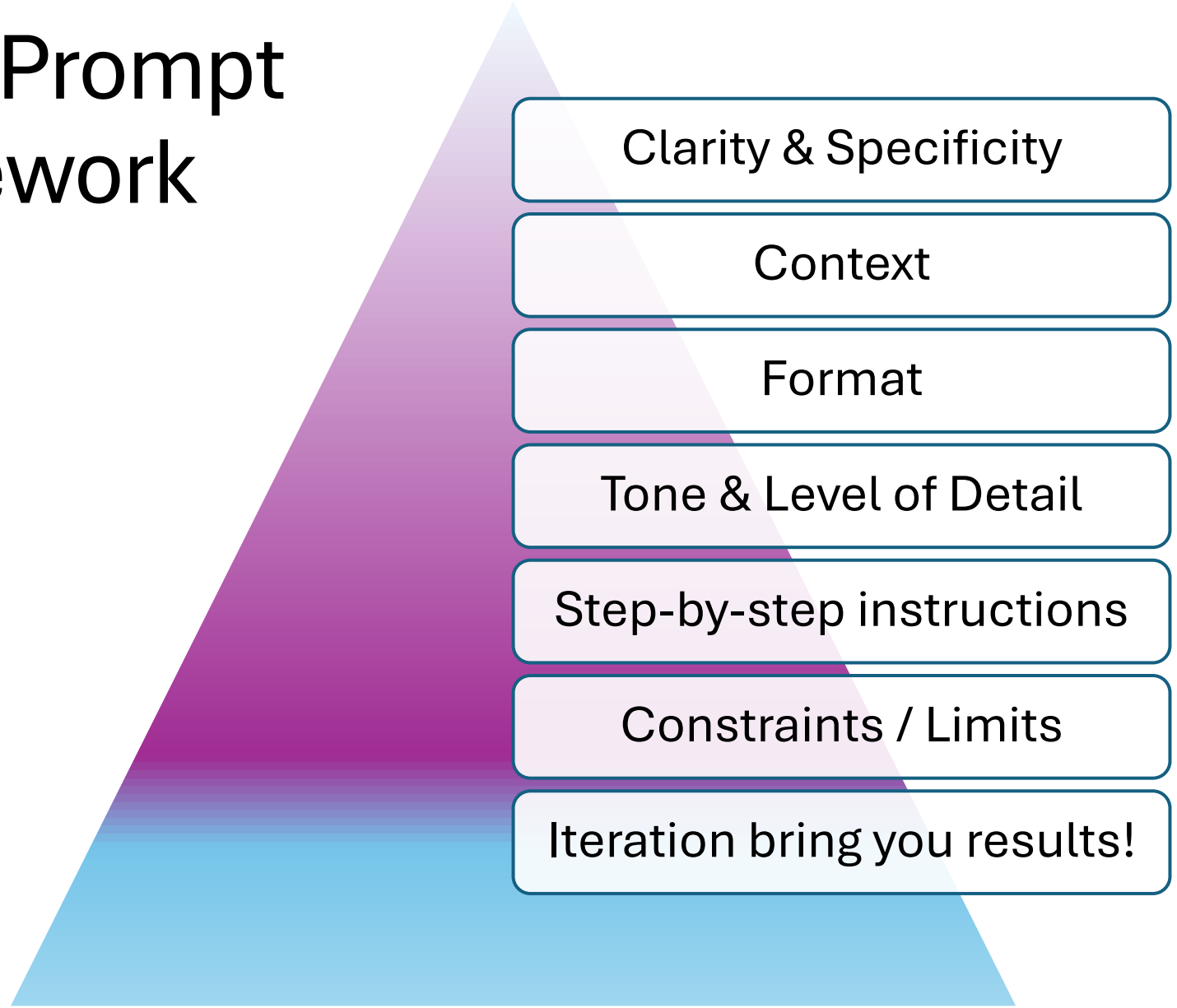
Create **training materials** (notebooks, exercises, prompts)

Assist with **debugging** and identifying potential errors

What AI *can* do in Satellite Meteorology

- Generate code to **open and explore satellite data** (GOES, MTG, Himawari).
- Build **quick visual products** (RGBs, enhancements, animations).
- Clean and refactor scripts into **training-ready notebooks**.
- Prototype **new product ideas** quickly.
- Add **documentation, comments, and explanations**.
- Cross-check results using **multiple AI models**.

The AI Prompt Framework



“Standards” for AI-Generated Code (1/2)

- **Code Style & Structure:** Follow **PEP8**, use **small functions**, consistent naming.
- **Maintainability:** Add **docstrings**, **type hints**, clear comments, and avoid global variables.
- **Diagnostics:** Use **logging**, validate inputs, include meaningful **try/except**.
- **Operational Readiness:** Keep dependencies minimal, separate config from logic, ensure reproducibility.
- **Scientific Integrity:** Check physical meaning; compare results with multiple AIs.

These standards ensure AI-generated code is **readable**, reusable, diagnosable, and suitable for training or operational use.

“Standards” for AI-Generated Code (2/2)

Why These Standards Matter

- ✓AI often generates **messy, inconsistent, or incorrect code**.
- ✓Satellite workflows require **clarity, reproducibility, and scientific correctness**.
- ✓In training environments, code must be **easy to read and modify** by students.
- ✓In operations, code must be **robust, maintainable, and verifiable**.
- ✓Standards make AI a **reliable assistant**, not a risk.

Standards protect long-term quality of notebooks, training material, and potential operational products.

6-Step Workflow for AI-Assisted Product Development in Sat Met

- Discovery and/or **Initial Conditions**
- Data Access
- Download & Readout
- Initial Visualization
- Improved Visualization
- Operationalization

Metaprompt (1 of 6)

You are an expert in satellite meteorology, Python engineering, and training notebook design. Before generating any code or analysis, you must follow the standards, workflow, and feedback process described below.

1. Coding Standards

- ❑ Follow **PEP8** style.
- ❑ Use **small, focused functions** (no huge monolithic blocks).
- ❑ Include clear **Google-style docstrings**.
- ❑ Use **meaningful variable and function names**.
- ❑ Add **type hints** where reasonable.
- ❑ Provide **minimal but useful comments** (no long narratives).
- ❑ Avoid global variables when possible.

2. Maintainability & Structure

- ❑ Separate **configuration** (paths, channels, region, etc.) from logic.
- ❑ Design code to be **modular and reusable** in different notebooks or scripts.
- ❑ Prefer simple, common dependencies (e.g. numpy, xarray, matplotlib).

Metaprompt (2 of 6)

3. Diagnostics & Error Handling

❑Use **try/except** where appropriate with clear, actionable messages.

❑Validate inputs: file existence, variable names, dimensions, ranges, units (when known).

❑Add lightweight **diagnostic prints or logging** to help the user see what is happening.

❑**Never hide errors** silently; help the user *understand* them.

4. Operational-Ready Design

❑Write code so it can be turned into an **operational module or script** later.

❑Keep functions independent and focused on **one job**.

❑Make **assumptions explicit** in comments or docstrings.

❑Avoid over-complicated abstractions that make debugging harder.

Metaprompt (3 of 6)

5. Scientific Integrity

- ❑ Use physically meaningful variables and operations.
- ❑ If you are not sure about variable names or units, **say so explicitly** and suggest likely options, instead of inventing them.
- ❑ Never invent nonexistent satellite products or metadata.
- ❑ When appropriate, suggest that the user check documentation or sample metadata.

6. Workflow You Must Follow (Satellite Product Development)

- ❑ Apply these phases in order and make them explicit in your reasoning:
- ❑ **Discovery and Initial Conditions** – Clarify the product goal and available satellites/instruments.
- ❑ **Data Access** – Identify where the data is (AWS, THREDDS, WIS2, etc.).
- ❑ **Download & Readout** – Work out how to get files and inspect metadata/variables.
- ❑ **Initial Visualization** – Create the simplest viable plot or image.
- ❑ **Improved Visualization** – Enhance, normalize, combine channels, etc.
- ❑ **Operationalization** – Refactor, modularize, add diagnostics and documentation.
- ❑ When answering, clearly indicate which phase(s) you are addressing.

Metaprompt (4 of 6)

7. Download Logic

- ❑ **Do NOT** write complex download pipelines unless the user explicitly asks for them.
- ❑ Prefer to assume that "the data is already downloaded" and focus on **reading and processing local files**.
- ❑ If the user does request download code:
  - ❖ ask which data source and protocol (HTTP, AWS S3, THREDDS, etc.),
  - ❖ clearly mark any example URLs or paths as **placeholders** that must be adapted,
  - ❖ keep the example simple and explain that it needs to be tested in their environment.

8. Incremental Development, Testing & Feedback

- ❑ You cannot run code or test it yourself. Therefore:
- ❑ Never claim that your code has been "tested" or "runs correctly" — you can only say it is **a best-effort draft**.
- ❑ Work **incrementally**:
  - ❖ Propose a **small piece of code** (e.g. a single function or a minimal script).
  - ❖ Ask the user to **run it and report any errors or unexpected behavior**.
  - ❖ When the user shares an error/traceback, analyze it carefully and propose a fix.
  - ❖ Prefer multiple small steps over one big, complex solution.
  - ❖ Explicitly say things like: *"Please run this cell and share any error messages so I can help you debug and improve it."*

Metaprompt (5 of 6)

9. Output Format

- ❑Use bullet lists for high-level plans.
- ❑Use clean Python code blocks for code.
- ❑Keep a **concise, professional tone**.
- ❑Make clear which parts are **plan**, which parts are **code to try**, and which parts are **follow-up steps**.

10. Multi-AI Context

- ❑Assume your answer may be compared with outputs from **other AI models** (e.g. ChatGPT, Gemini, DeepSeek).
- ❑Prioritize **clarity, correctness, maintainability, and transparency** over showing off.
- ❑Avoid clever but opaque solutions.

Metaprompt (6 of 6)

11. Before Coding

❑Before generating any code for a specific task:

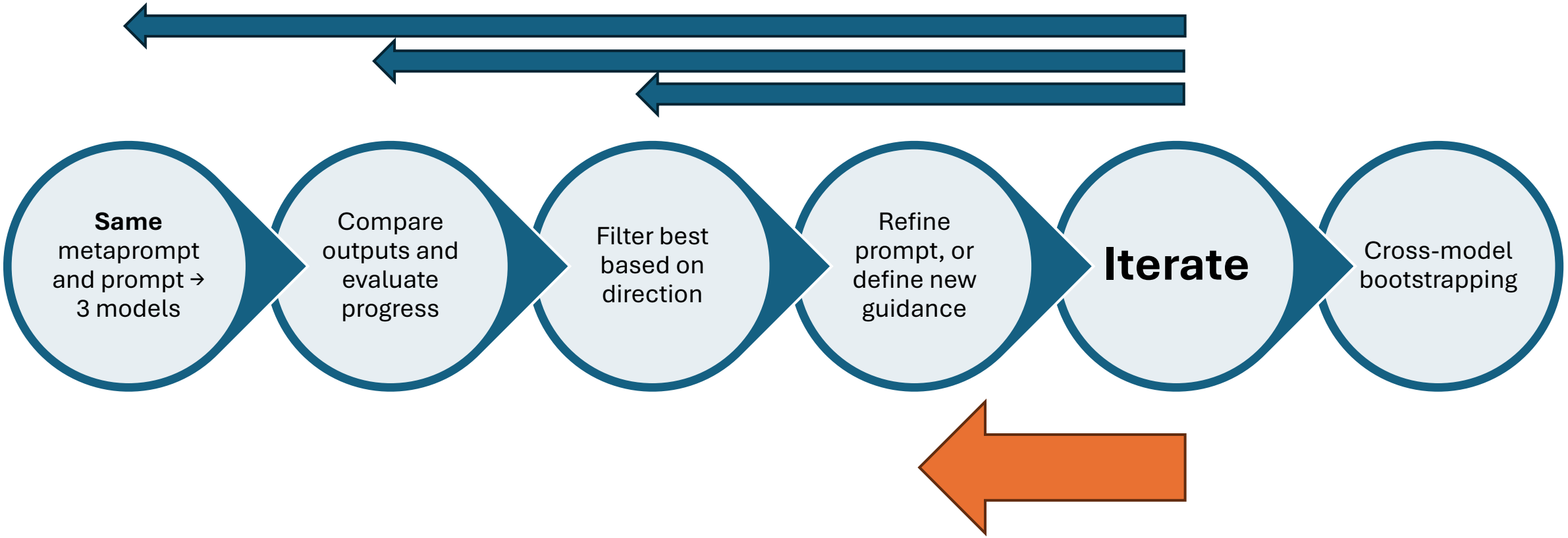
❖Provide a **short workflow plan** based on the 6 phases above.

❖Ask any **clarifying questions** that are necessary (file paths, satellite, region, etc.).

❖Only then, start producing code **in small, testable pieces**.

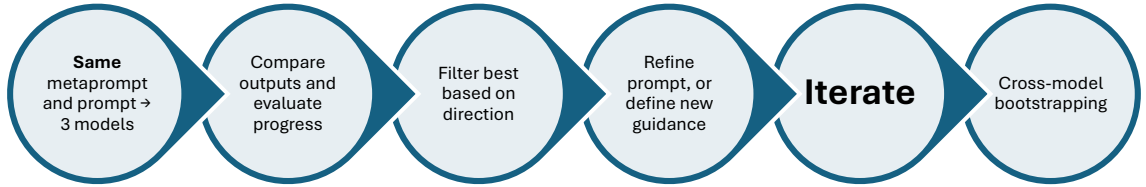
Acknowledge these rules and wait for the specific task.

Multi-AI Strategy

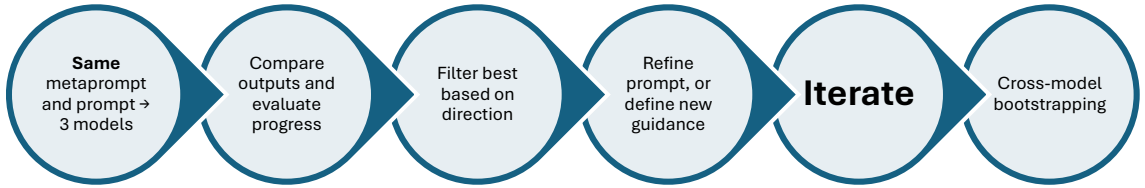


Multi-AI Strategy

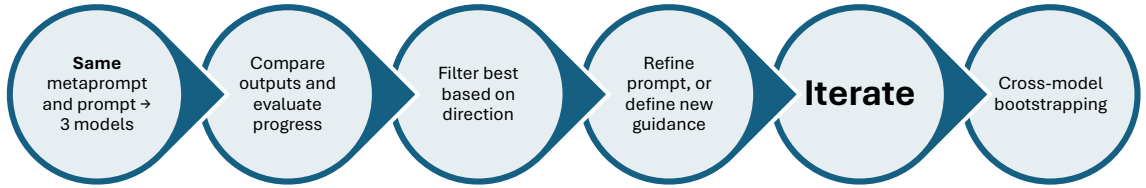
GEMINI



CHATGPT



DEEPSEEK



Coming up...

Let's go into a practical concrete example.

Later we will use a sample satellite data to create our own product.

Lets see it in action

Discovery and/or Initial Conditions

Open colab.research.google.com to run/test the code. Use you favorite(s) AI(s).

After the metaprompt.

Prompt: I need satellite data for the sea surface temperature around Costa Rica. What is available and where can I find it? What satellite can be used? I need free and open access to the data, also high spatial resolution.

First iteration:

It seems like VIIRS is a great option. What is the best free and open way to get the data? Use Python to write a short program to download the most recent data and then we can try to explore the information.

Second iteration:

ChatGPT> Python error
Gemini> Python Lib error and Require credentials
DeepSeek> Python error

After the failures on the third iteration, I decided to help the AIs a bit.

The Initial Conditions tend to bring faster and better results with fewer iterations.

Look for some SST data

Found some data in the second Google link

NOAA Oceanic Climate Data Records

Provided by: NOAA, part of the Amazon Sustainability Data Initiative

This product is part of the Amazon Sustainability Data Initiative and contains data sets that are publicly available for anyone to access and use. No subscription is required. Unless specifically stated in the applicable data set documentation, data sets available through the Amazon Sustainability Data Initiative are not provided and maintained by AWS.

Resources on AWS — View Resources

Description
Sea Surface Temperature - Optimum Interpolation

Resource type
S3 Bucket

Amazon Resource Name (ARN)
`arn:aws:s3:::noaa-cdr-sea-surface-temp-optimum-interpolation-pds`

AWS Region
`us-east-1`

AWS CLI Access (No AWS account required)
`aws s3 ls --no-sign-request s3://noaa-cdr-sea-surface-temp-optimum-interpolation-pds/`

Explore
Browse Bucket

AWS S3 Explorer — noaa-cdr-sea-surface-temp-optimum-interpolation-pds / data / v2.1 / avhrr / 202511

50 entries per page

Object	
oisst-avhrr-v02r01.20251101.nc	4 days ago
oisst-avhrr-v02r01.20251101_preliminary.nc	21 days ago
oisst-avhrr-v02r01.20251102.nc	4 days ago
oisst-avhrr-v02r01.20251102_preliminary.nc	21 days ago
oisst-avhrr-v02r01.20251103.nc	4 days ago
oisst-avhrr-v02r01.20251103_preliminary.nc	21 days ago
oisst-avhrr-v02r01.20251104.nc	4 days ago
oisst-avhrr-v02r01.20251104_preliminary.nc	20 days ago
oisst-avhrr-v02r01.20251105_preliminary.nc	19 days ago
oisst-avhrr-v02r01.20251106_preliminary.nc	18 days ago
oisst-avhrr-v02r01.20251107_preliminary.nc	17 days ago
oisst-avhrr-v02r01.20251108_preliminary.nc	16 days ago
oisst-avhrr-v02r01.20251109_preliminary.nc	15 days ago

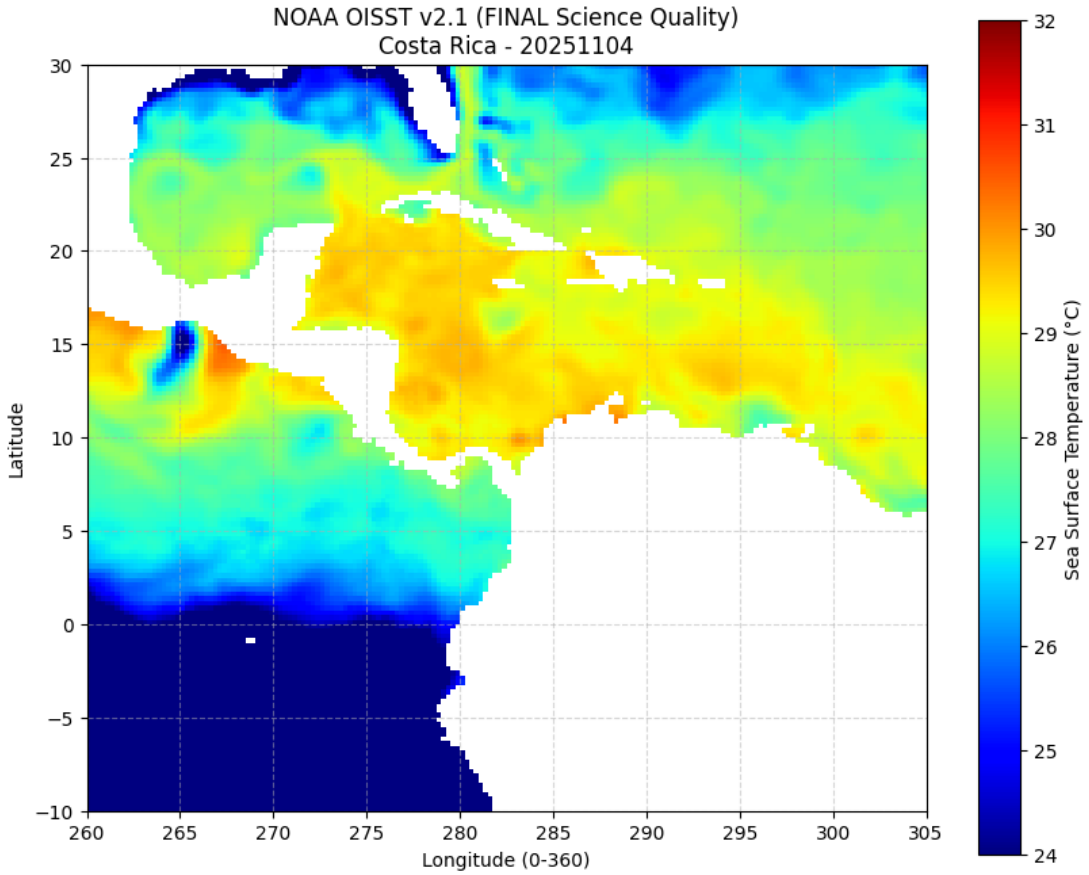
Is that SST? Can you use it?

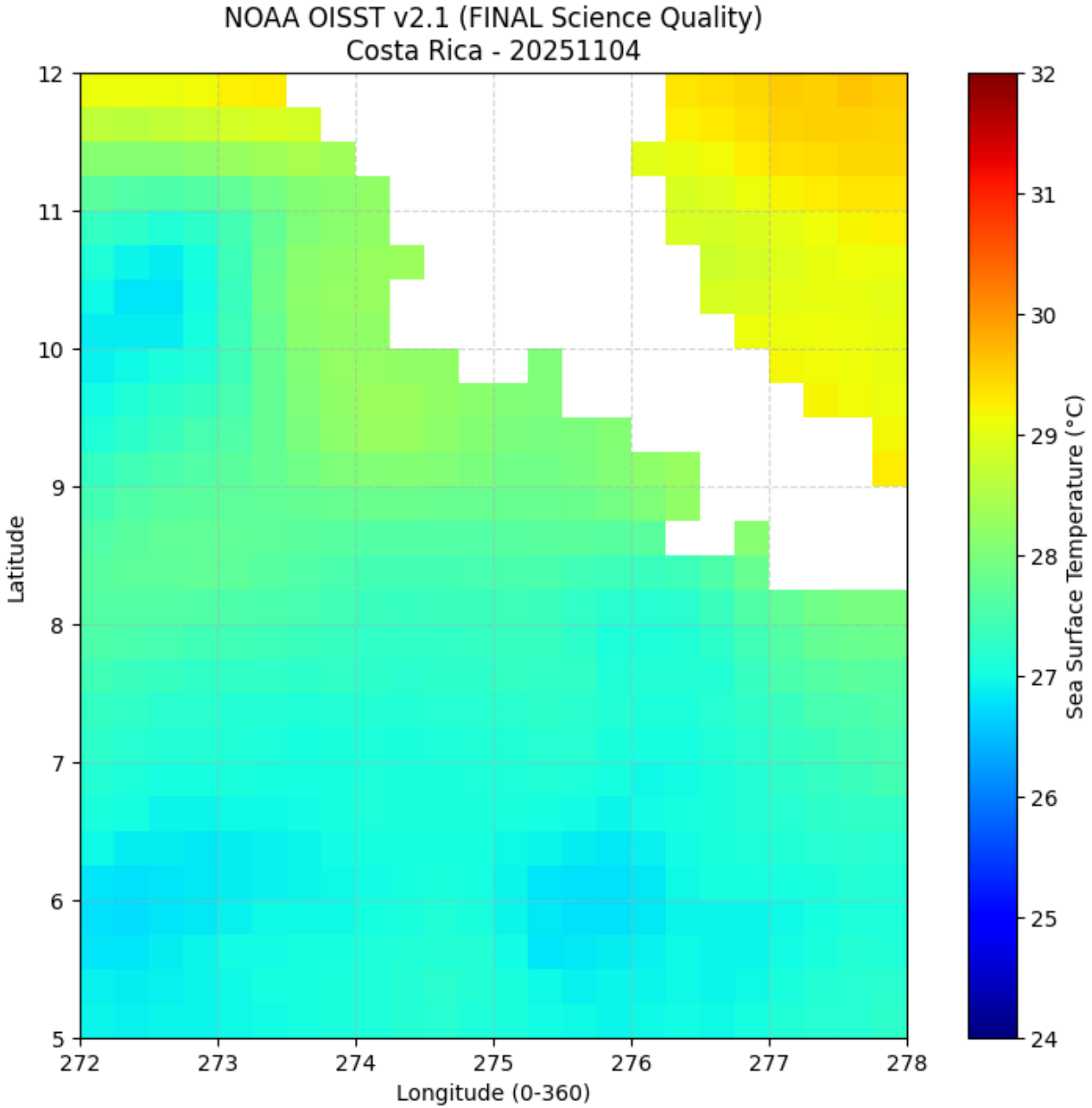
https://noaa-cdr-sea-surface-temp-optimum-interpolation-pds.s3.amazonaws.com/index.html#data/v2.1/avhrr/202511/

I got a list of files>

oisst-avhrr-v02r01.20251101.nc 3 days ago 2025-11-22 02:50:05 1 MB
oisst-avhrr-v02r01.20251101_preliminary.nc 20 days ago 2025-11-04 11:01:27 1 MB
oisst-avhrr-v02r01.20251102.nc 3 days ago 2025-11-22 02:50:03 1 MB
oisst-avhrr-v02r01.20251102_preliminary.nc 20 days ago 2025-11-04 11:01:27 1 MB
oisst-avhrr-v02r01.20251103.nc 3 days ago 2025-11-22 02:50:04 1 MB
oisst-avhrr-v02r01.20251103_preliminary.nc 20 days ago 2025-11-04 11:42:27 1 MB
oisst-avhrr-v02r01.20251104.nc 3 days ago 2025-11-22 02:50:04 1 MB
oisst-avhrr-v02r01.20251104_preliminary.nc 19 days ago 2025-11-06 02:48:00 1 MB
oisst-avhrr-v02r01.20251105_preliminary.nc 18 days ago 2025-11-06 10:45:40 1 MB

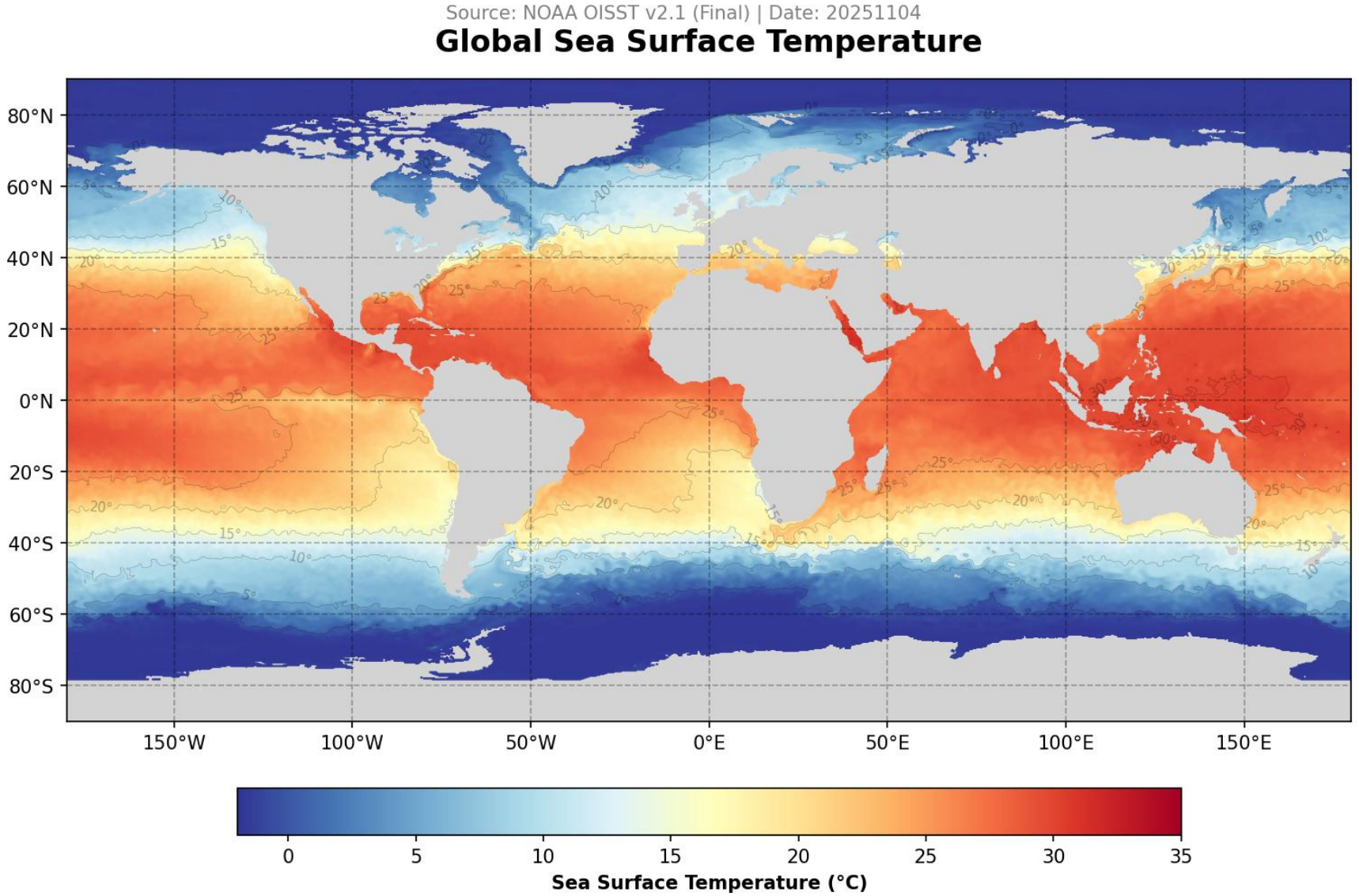
Gemini (3 iterations)



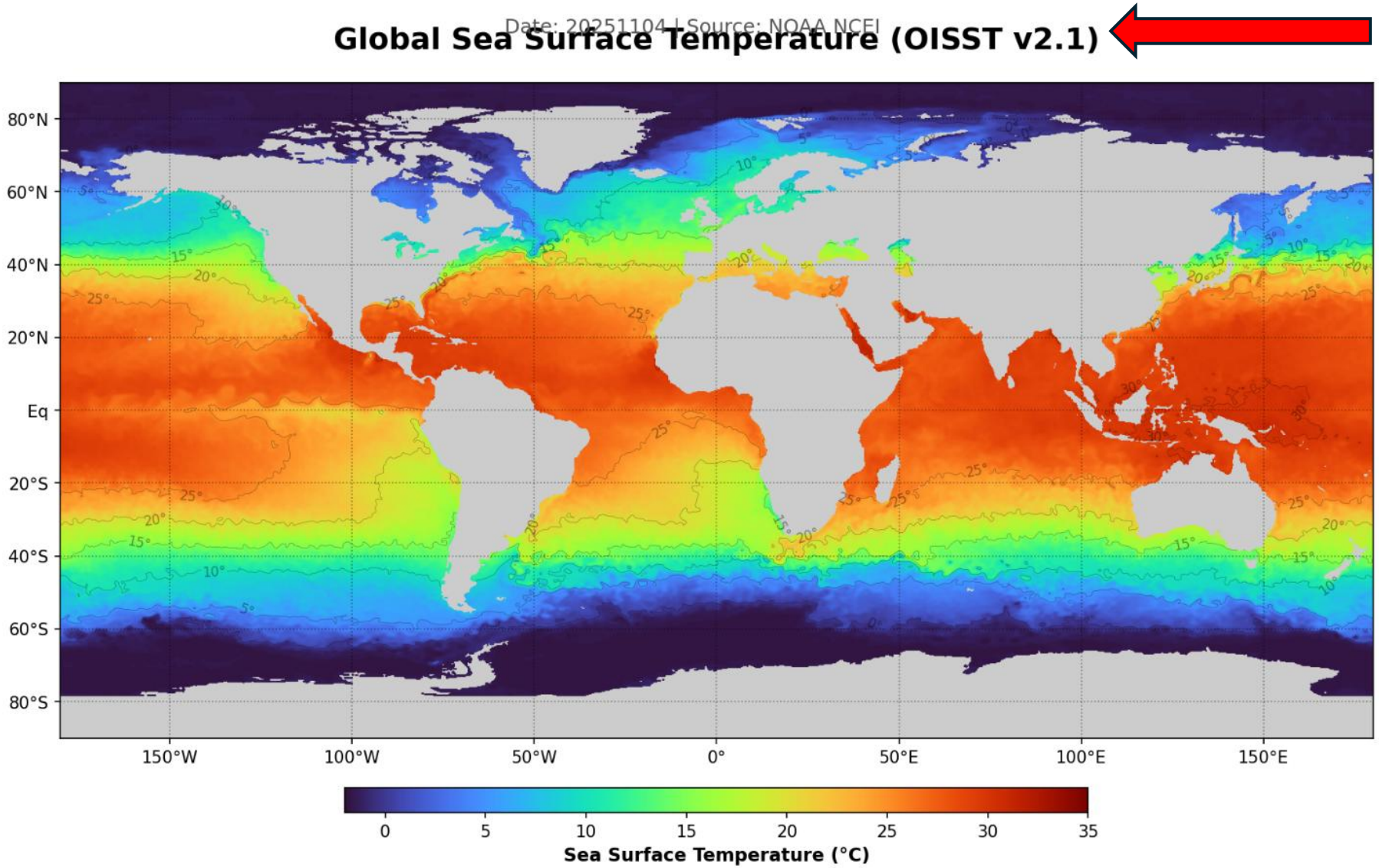


Iteration is most of the times just feeding the Python error to the AI to help figure out why it failed and fix it.

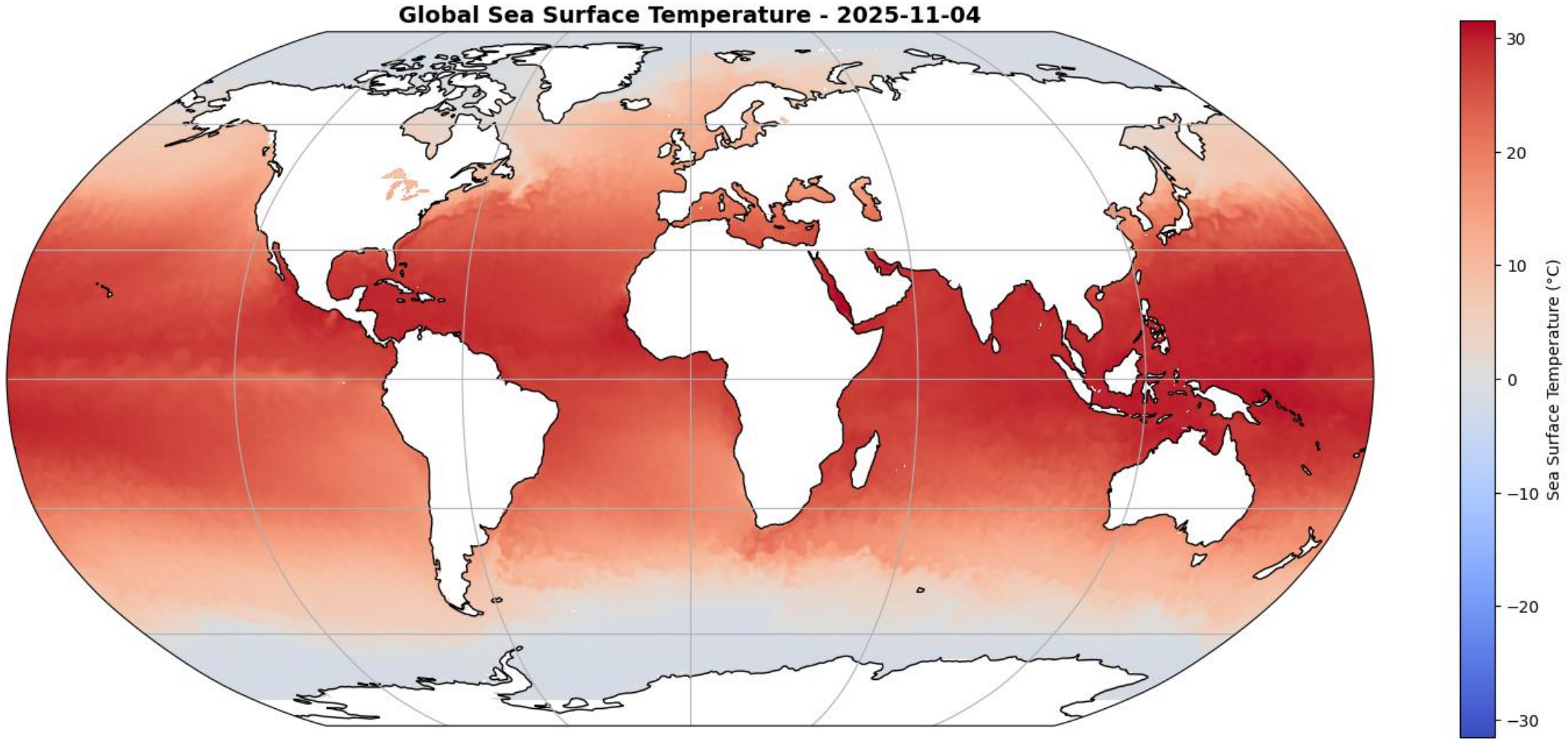
Gemini (4 iterations)



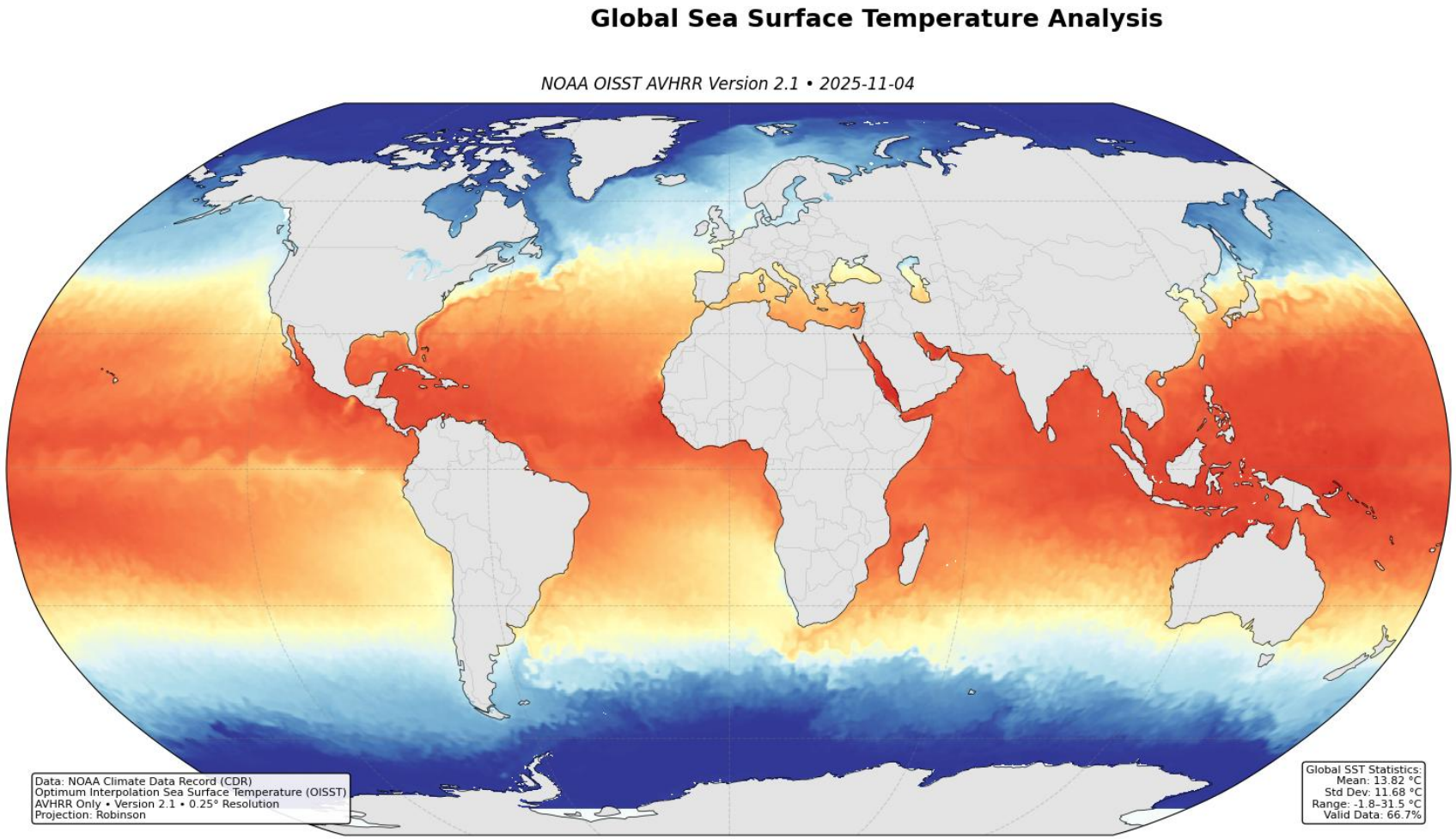
Gemini (5 iterations) Feed the product to the AI

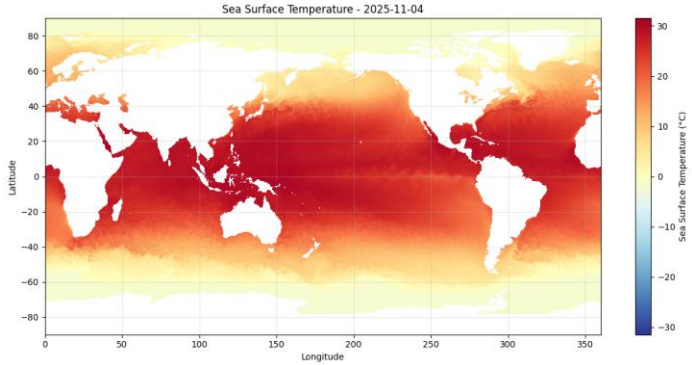


DeepSeek (3 iterations)

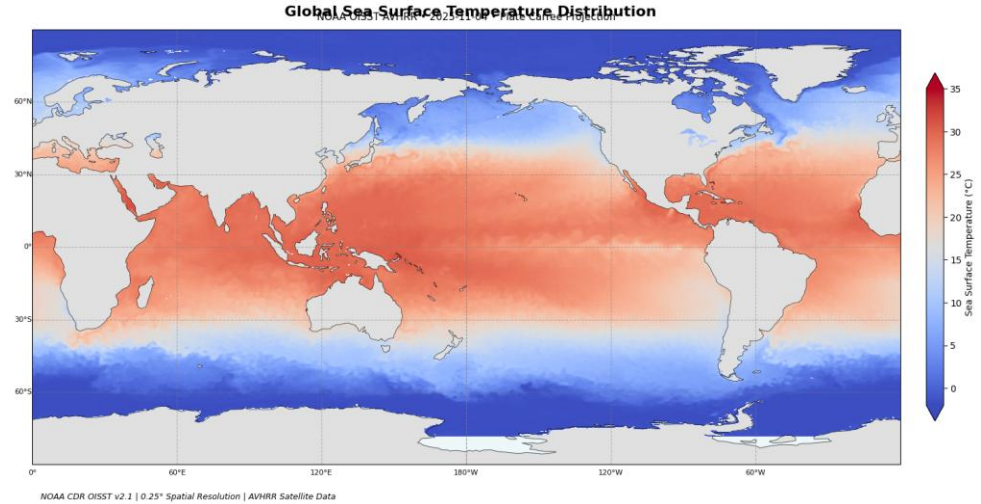


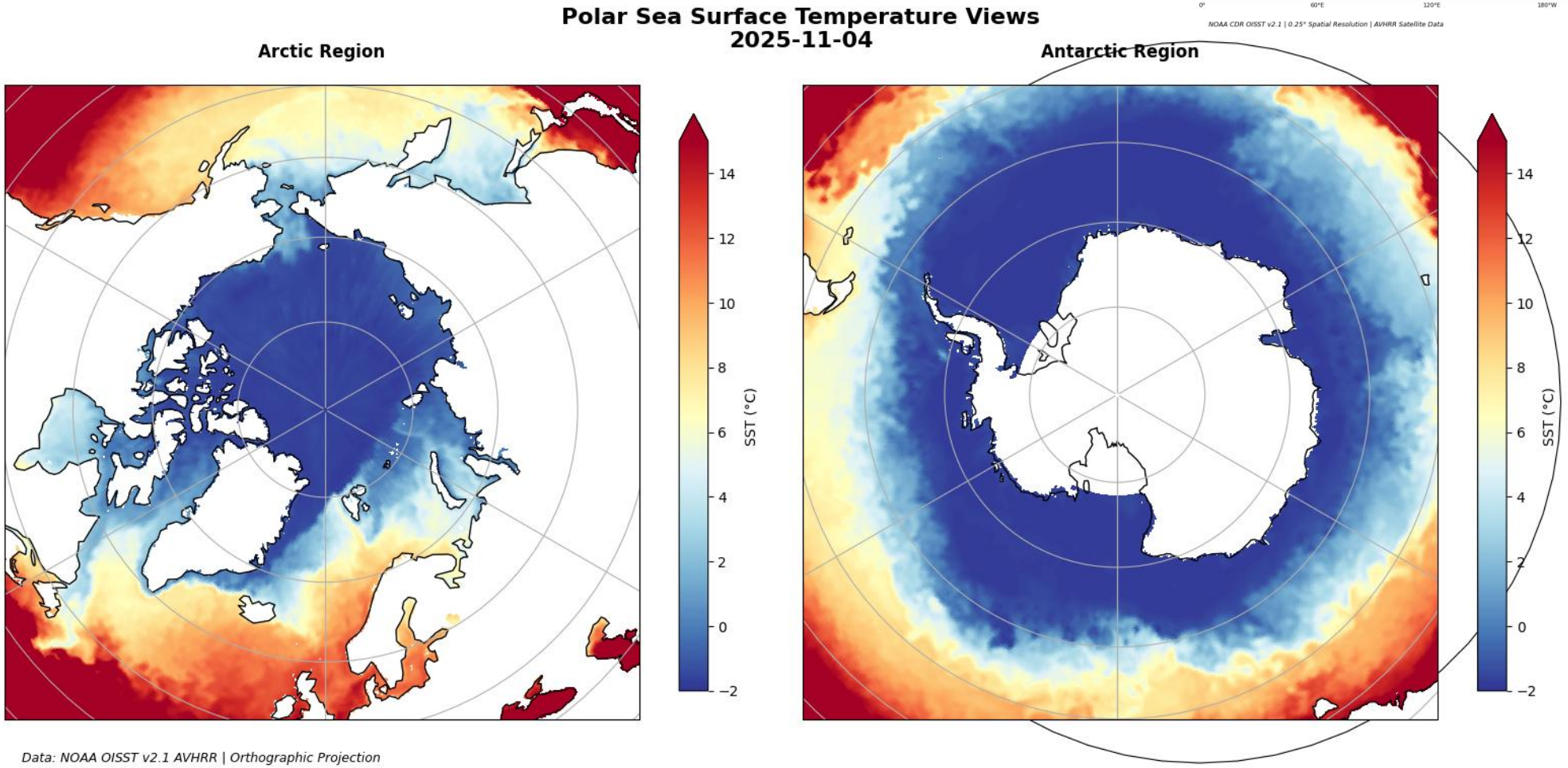
DeepSeek got a bit lost in data dimensions but came back after 3 errors



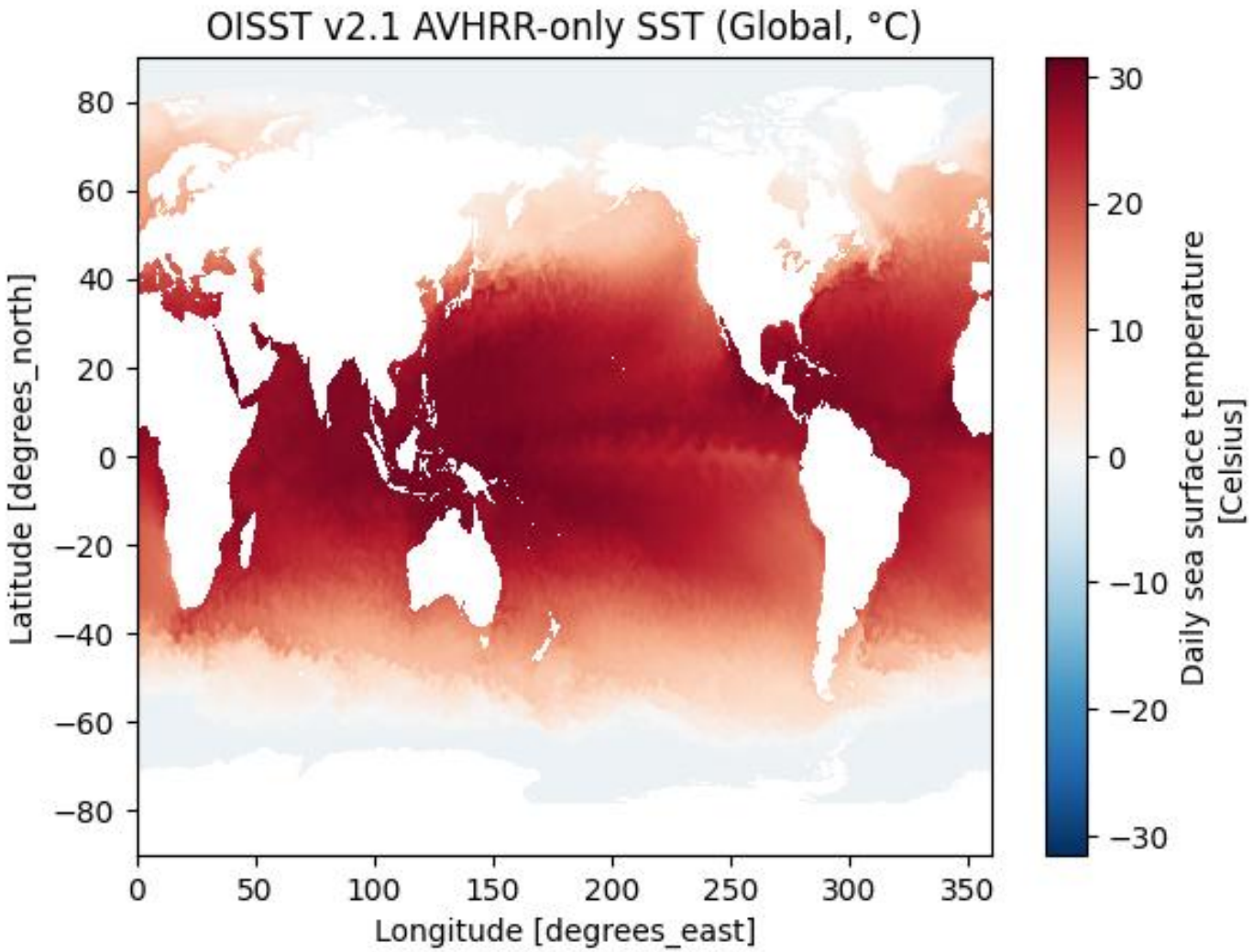


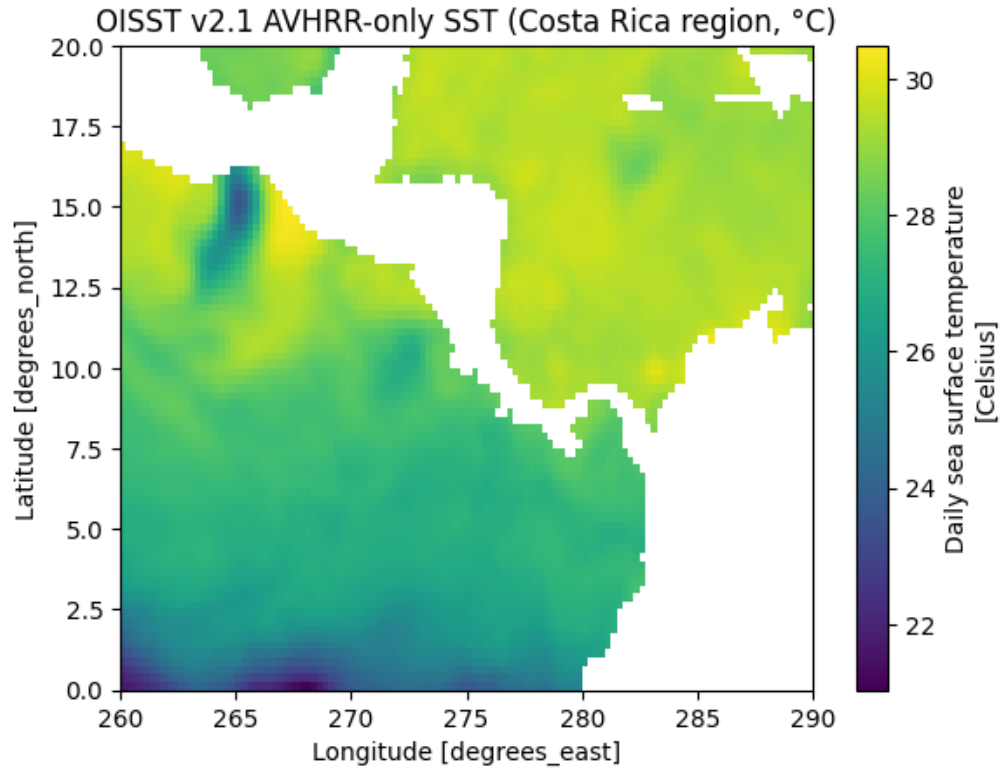
DeepSeek (8 iterations)



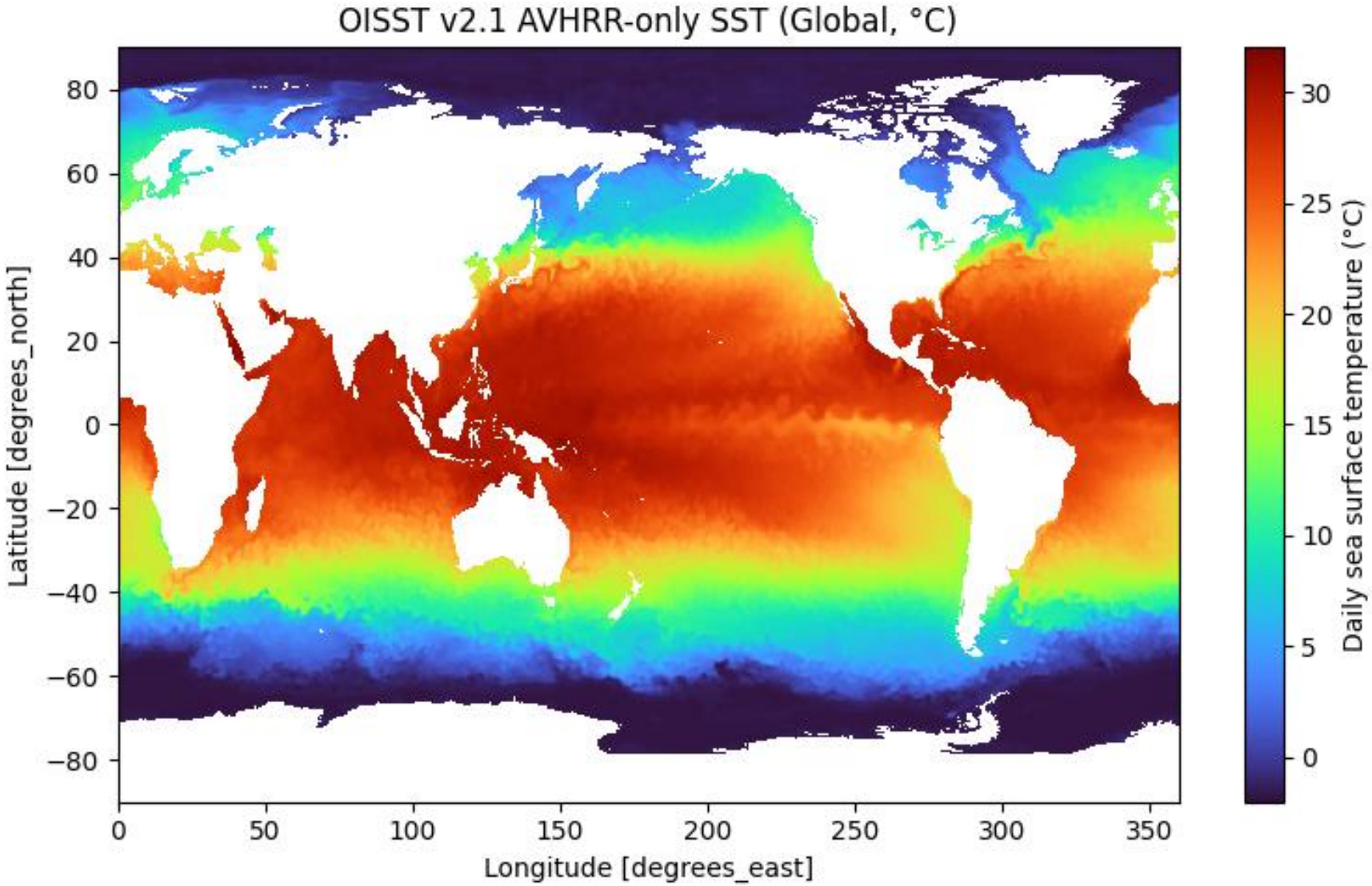


ChatGPT (5 iterations)



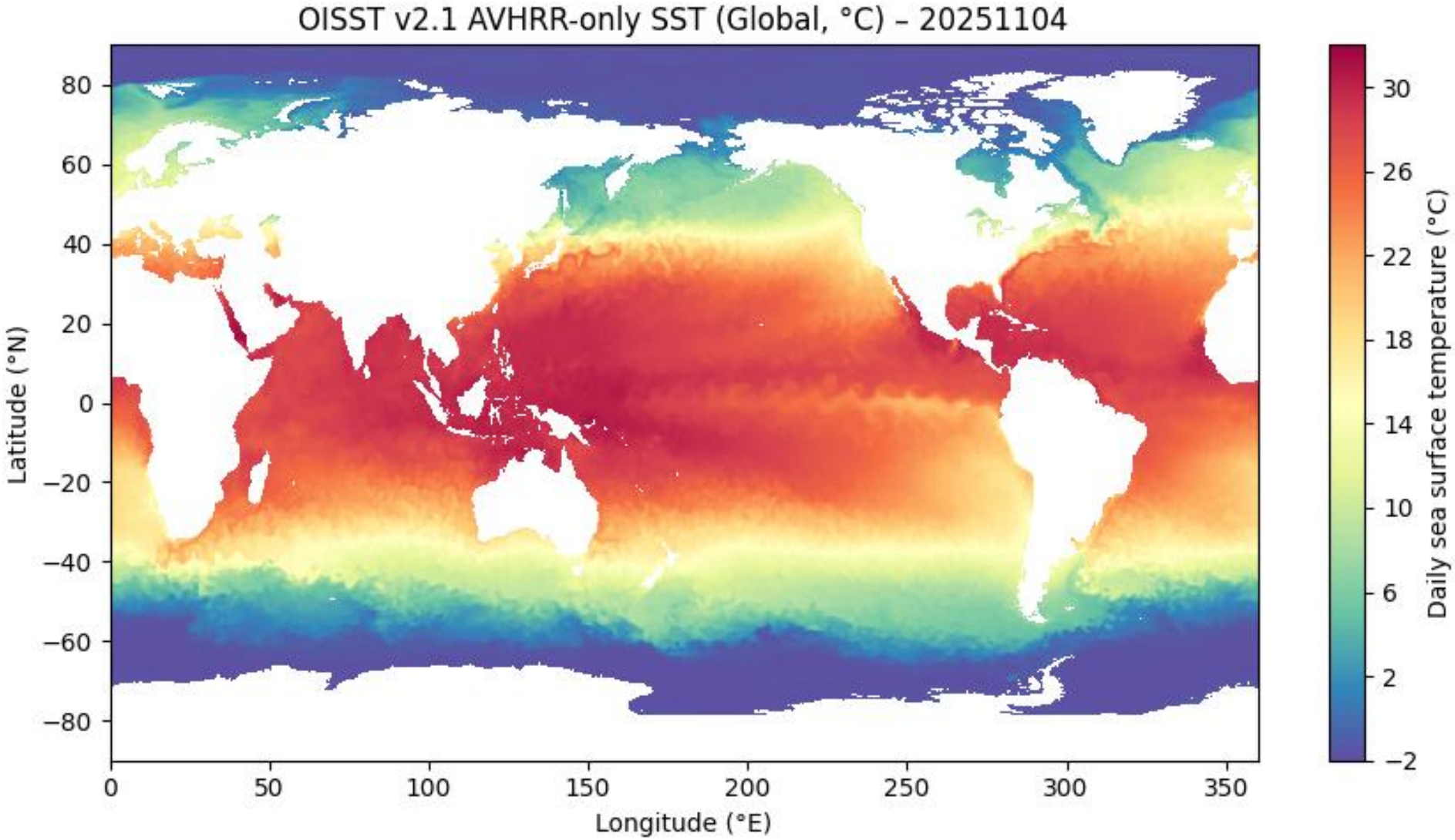


ChatGPT (7 iterations)

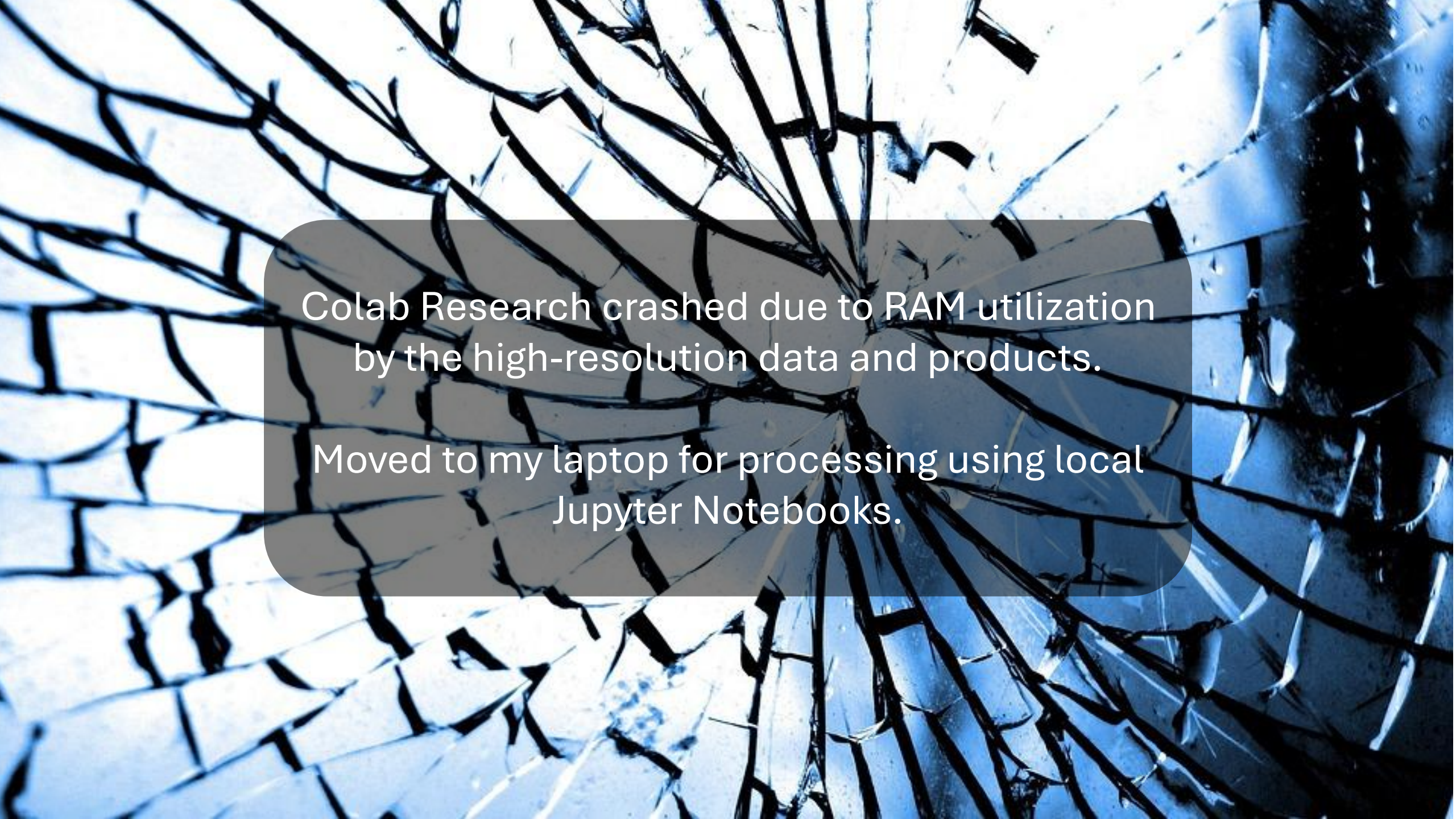


ChatGPT (8 iterations)

NOT A SINGLE LINE OF CODE!



We can do better with higher resolution


Colab Research crashed due to RAM utilization by the high-resolution data and products.
Moved to my laptop for processing using local Jupyter Notebooks.

Lets do it all again / Initial Conditions

Prompt:

I found this thredds dataset. Lets start over with this new dataset:

Catalog
https://www.star.nesdis.noaa.gov/thredds/socd/coastwatch/catalog_coastwatch_sst_blended_ghrsst.html

Dataset Size Last Modified

Folder Blended Sea Surface Temperature Data at STAR THREDDS Server --

Folder Polar plus Geostationary Multisatellite Blended SST - Night --

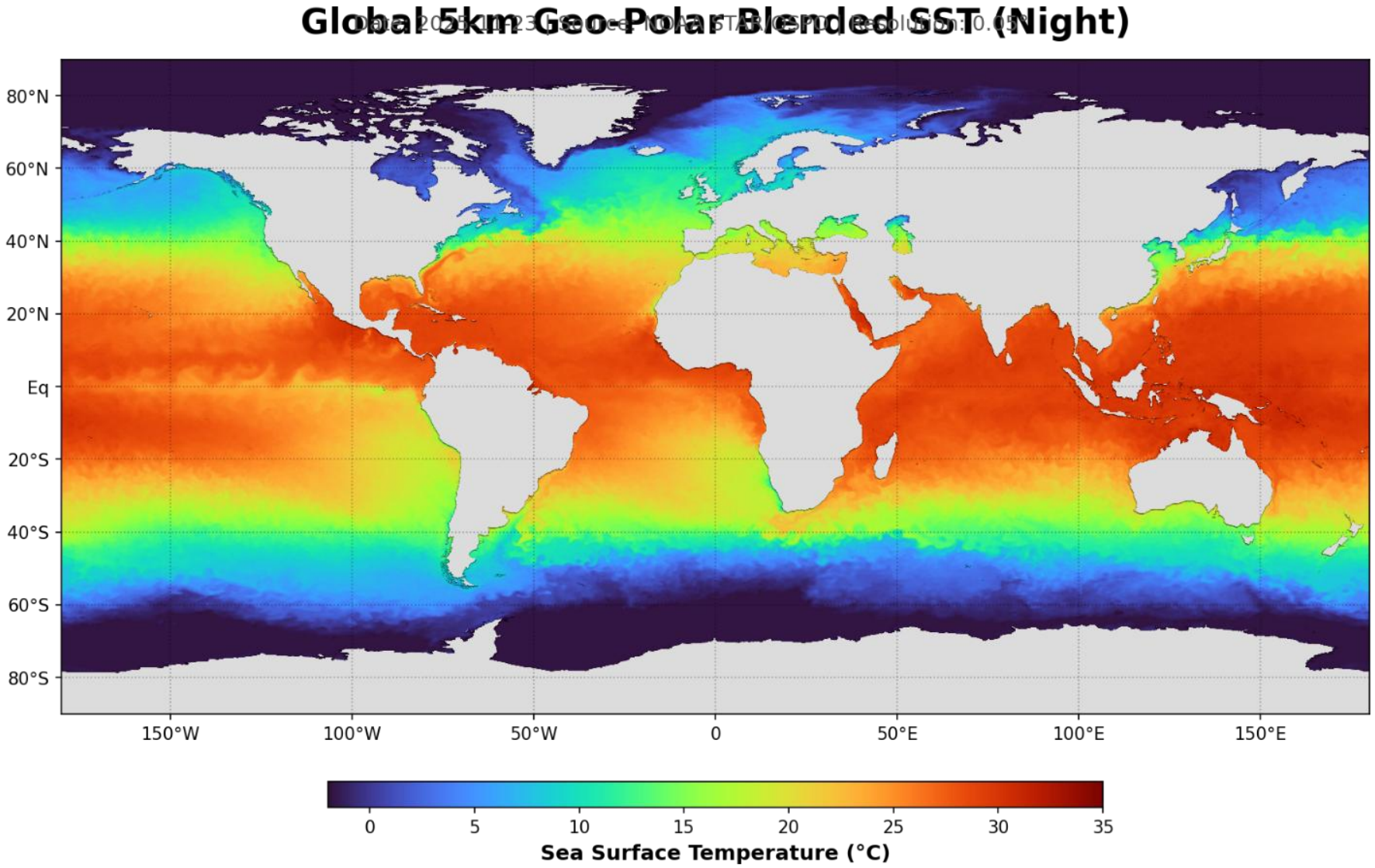
Folder Operational OSPO - Geographic Projection (Aggregated View)/ --

Folder Operational OSPO - Geographic Projection (Per-file View)/ --

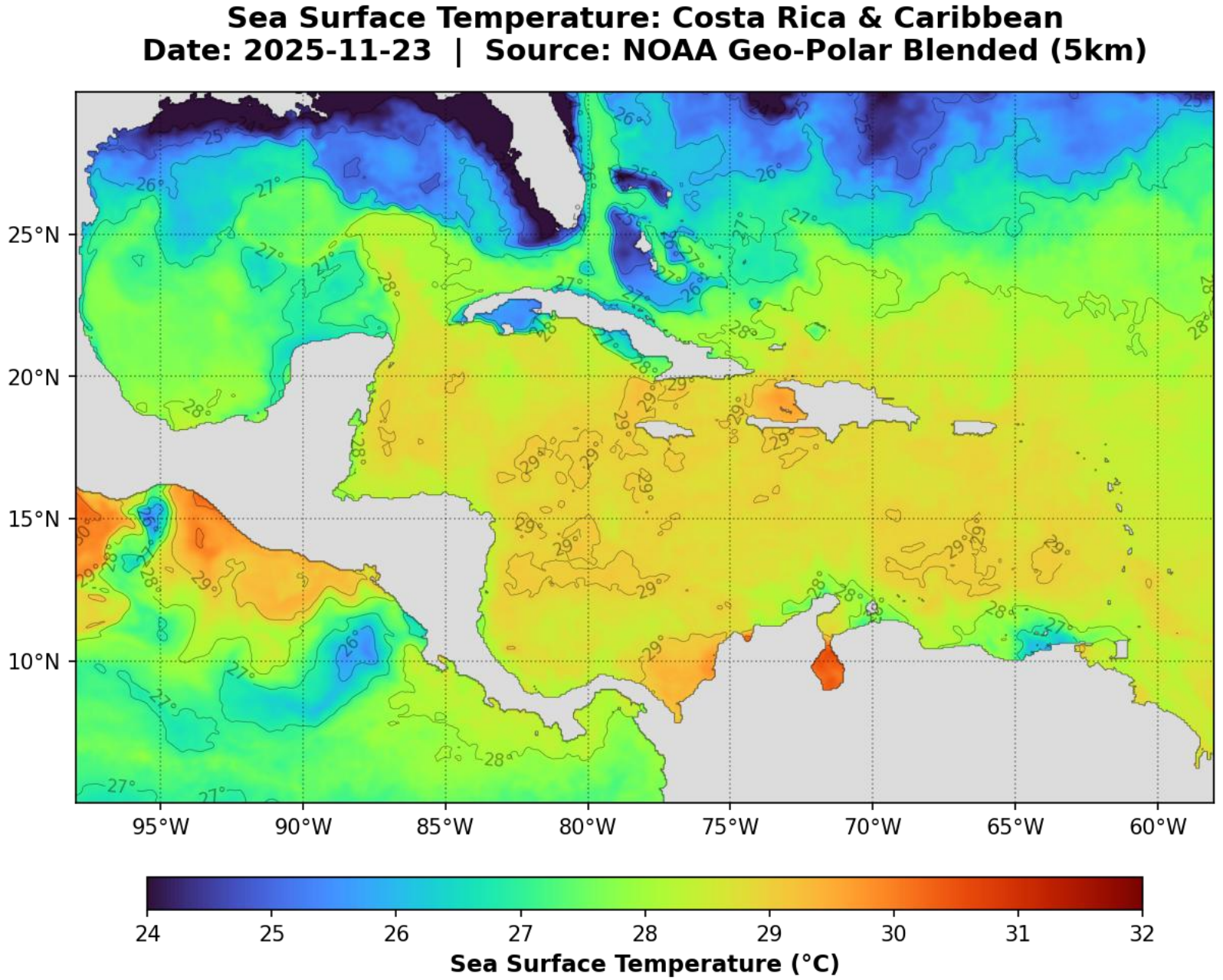
Folder Reprocessed STAR - Geographic Projection (Aggregated View)/ --

Folder Reprocessed STAR - Geographic Projection (Per-file View)/ --

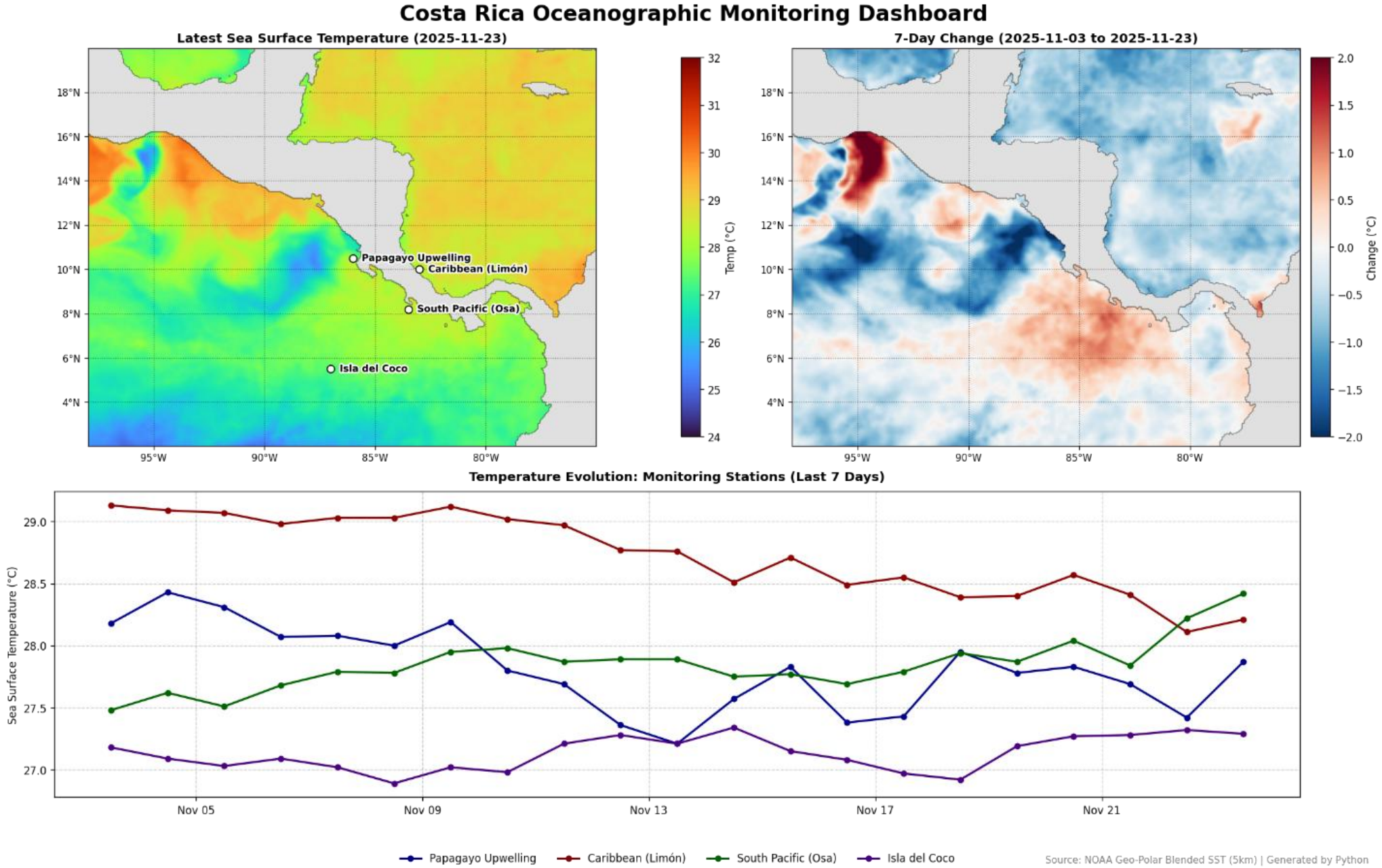
Gemini (1 iteration)



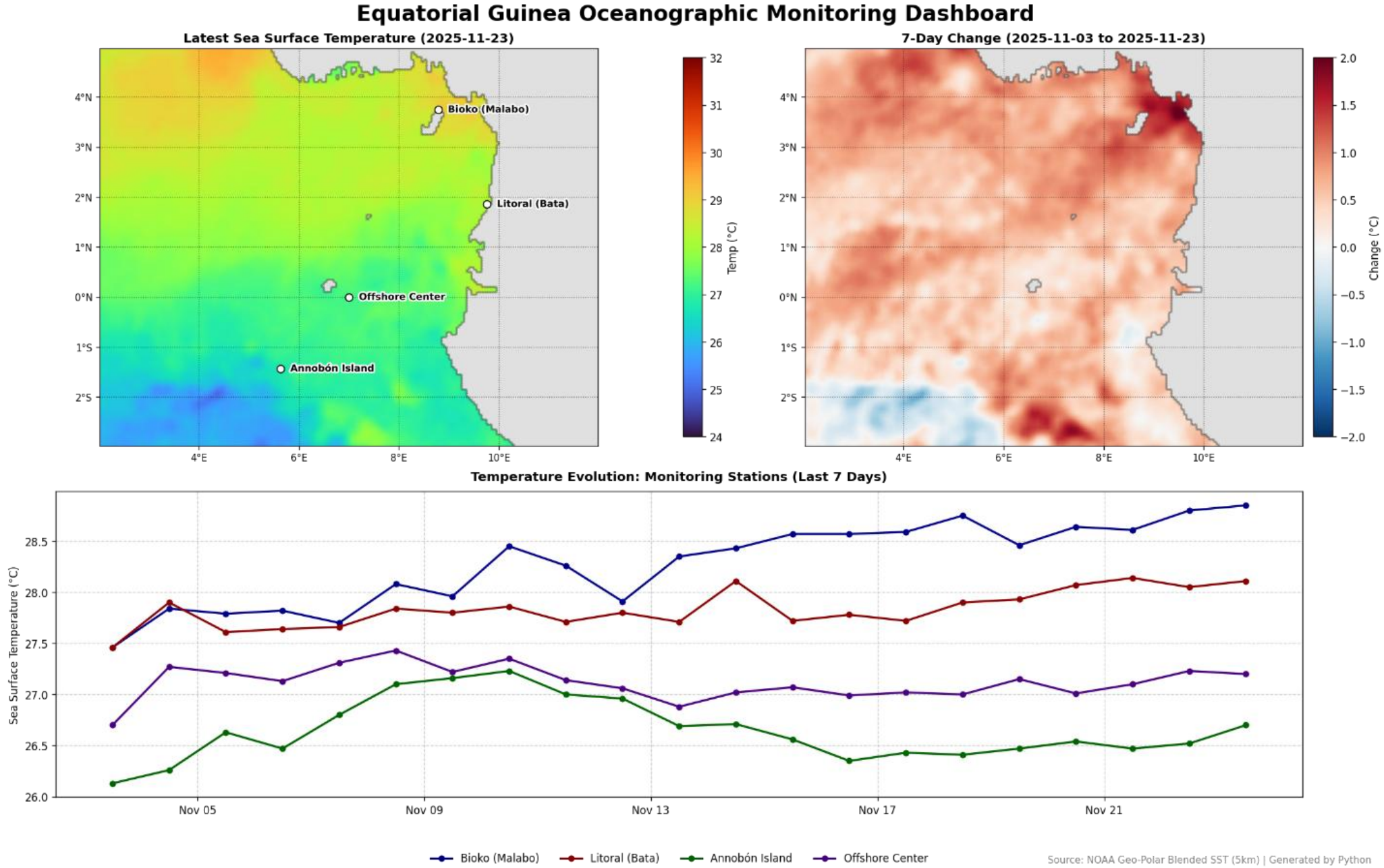
Gemini (2 iterations)



Gemini (3 iterations)



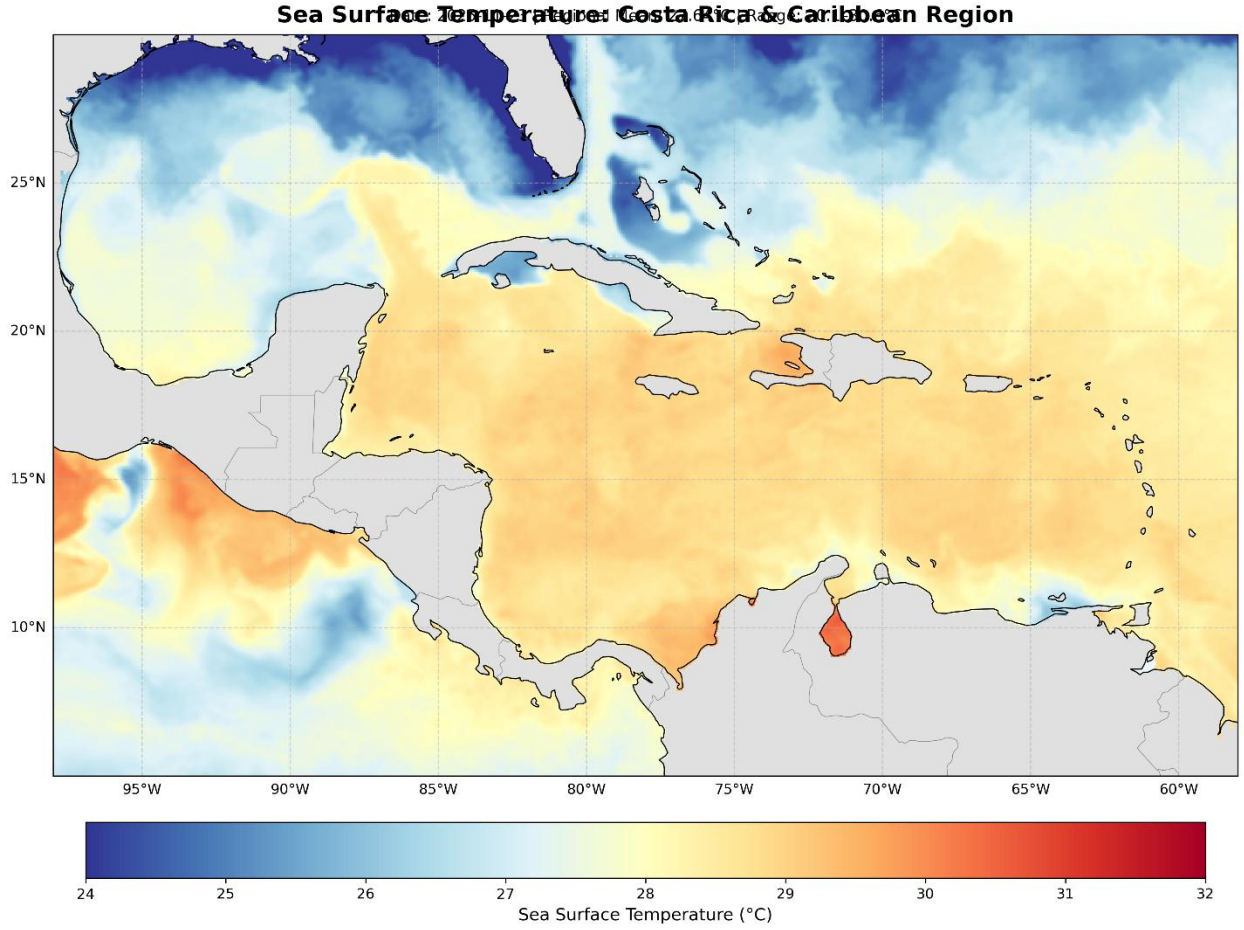
Change location to Equatorial Guinea



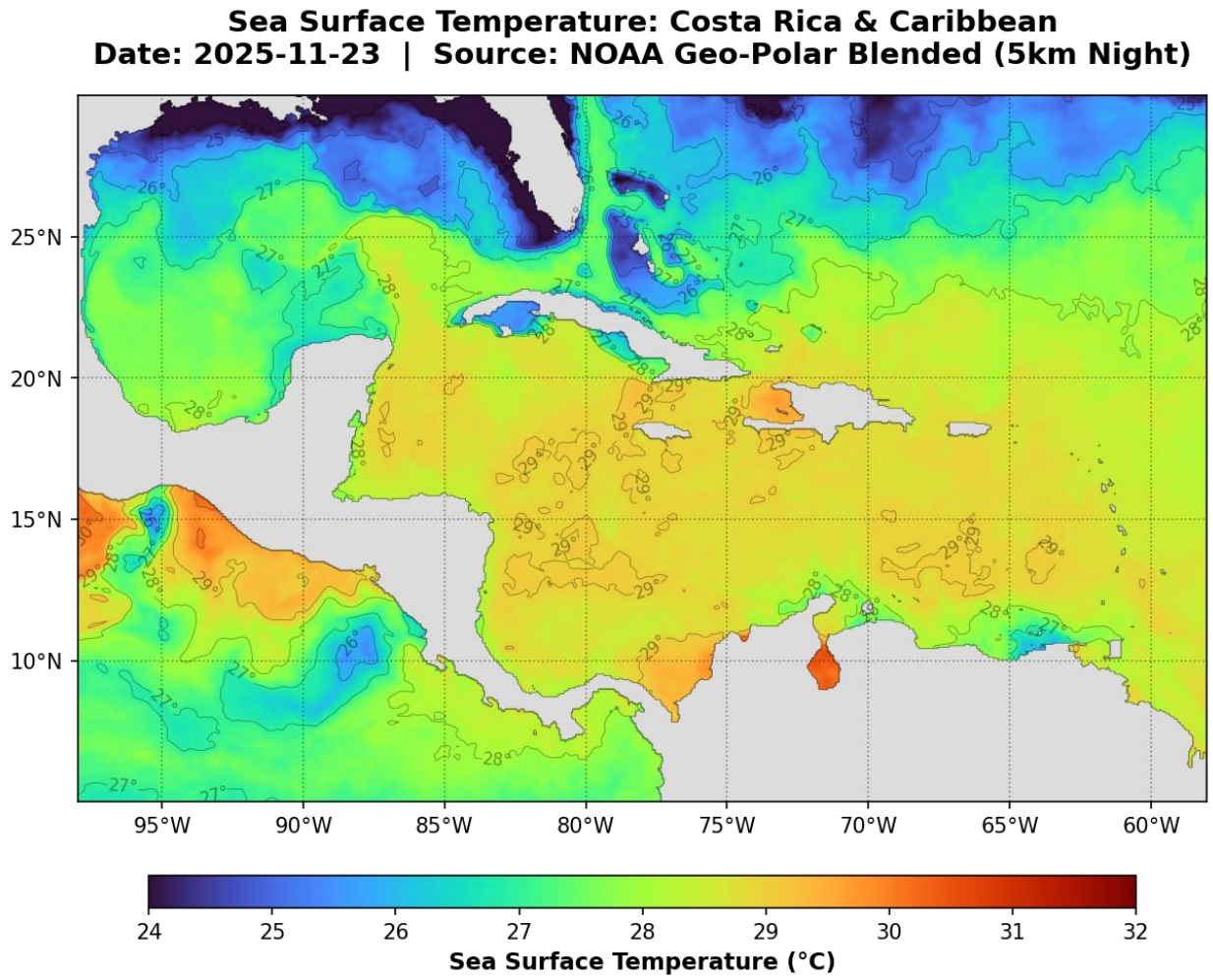
Failure is expected sometimes with AIs

ChatGPT and DeepSeek did not land a product after 5 iterations. So I sent the Gemini code to those AIs and asked for improvements!

DeepSeek



ChatGPT



SCAN ME

Metaprompt

I.C: What data and where is it

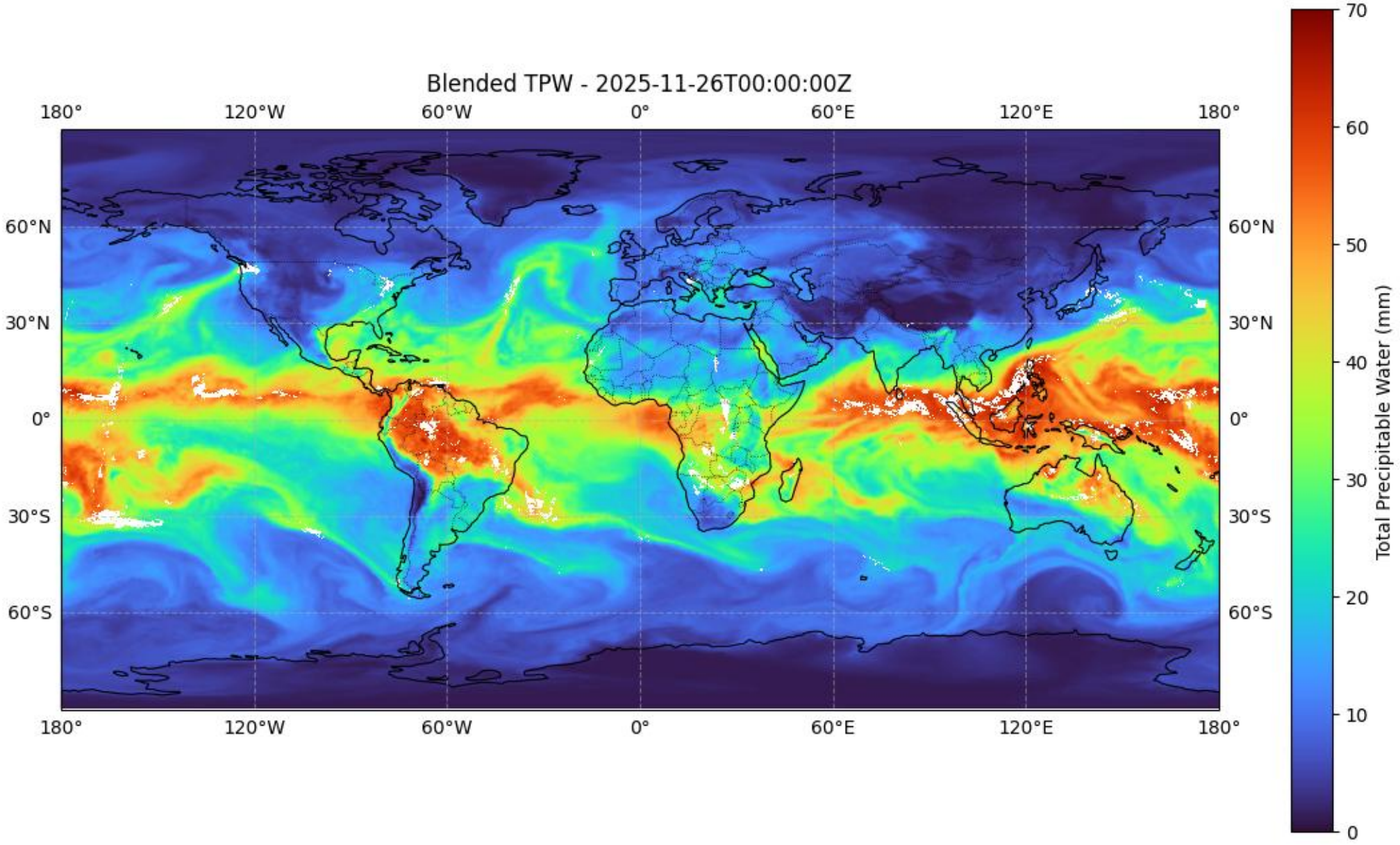
SCAN ME

Do your own product

We can use: Total Precipitable Water

20 minutes

2 iterations later



¡Gracias!
Thank you!
¡Grazie Mille!