

Seasonal sea level forecasts are useful in the tropical Pacific

Matthew Widlansky
mwidlans@hawaii.edu

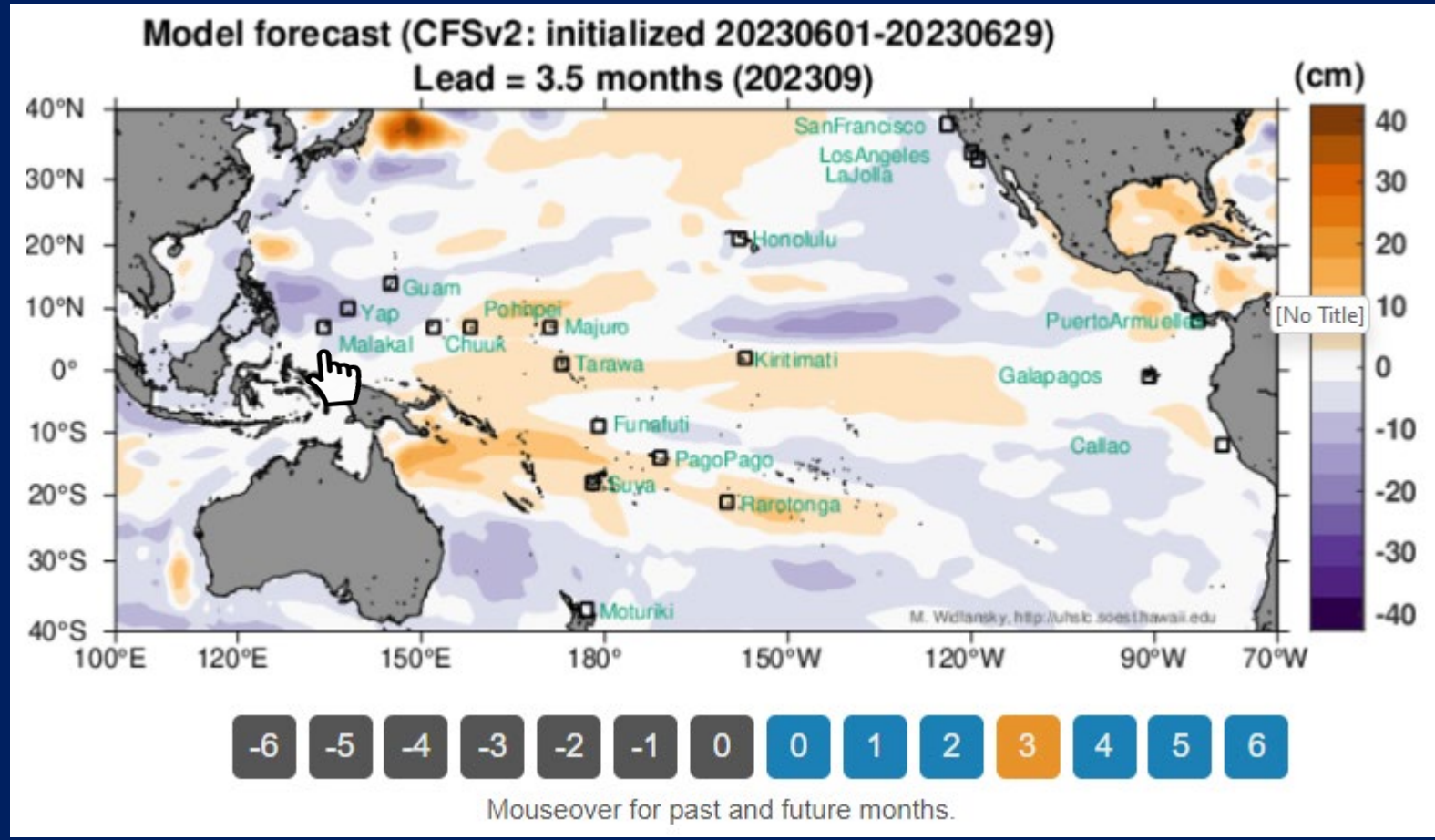
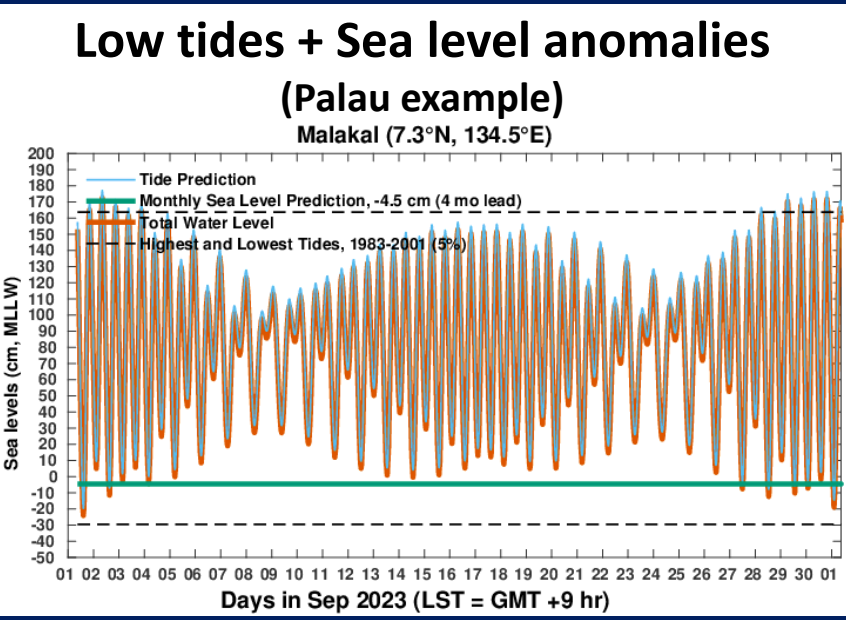


<http://uhslc.soest.hawaii.edu/sea-level-forecasts/>

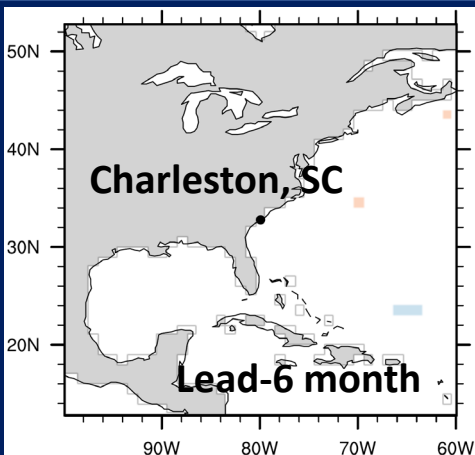
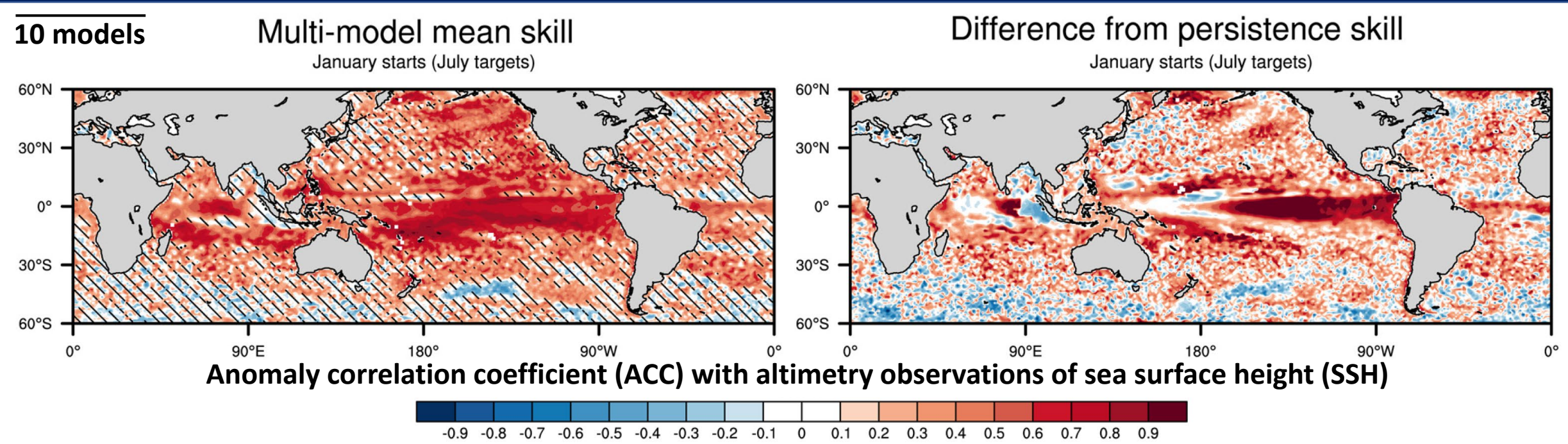
Product features

- 1) Forecast discussion
- 2) Island outlooks
- 3) Low or high-water alerts

September 2023 forecast of sea level anomalies



Climate forecast systems are skillful in most of the tropics, but perform poorly at higher latitudes and along some coasts

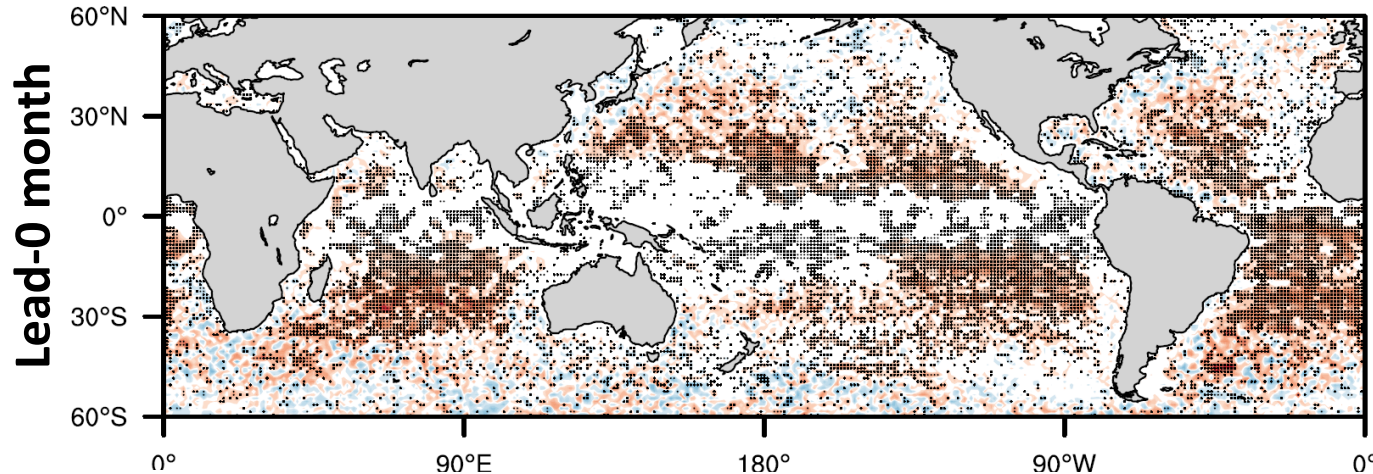


Problem:
Regional forecasts are not significantly correlated with East Coast tide gauge observations

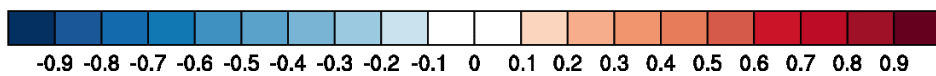
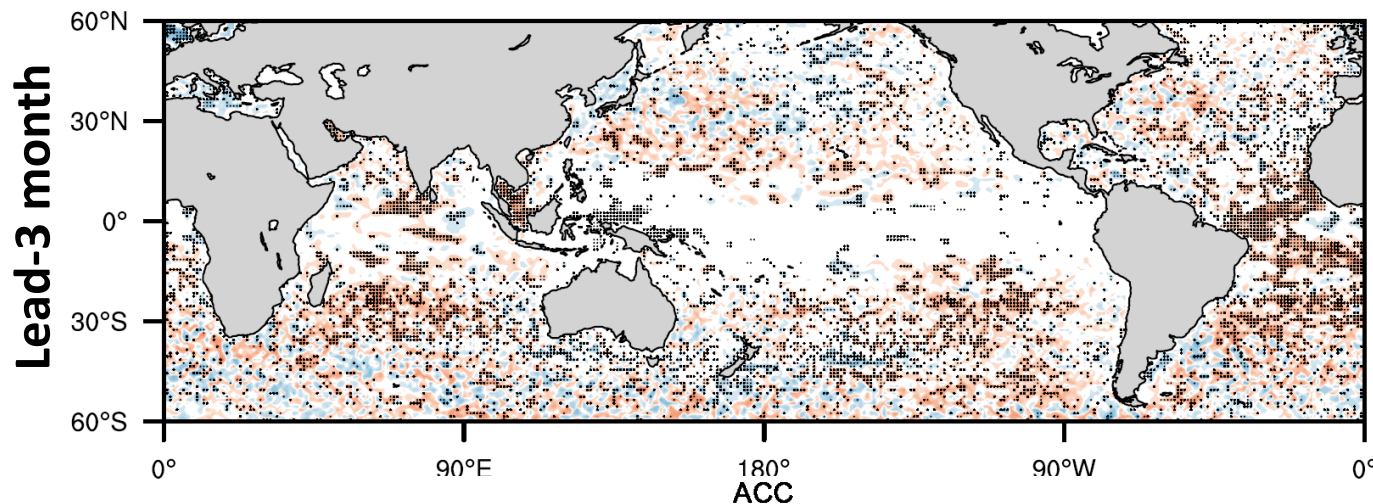
Long et al. 2021 "Seasonal forecasting skill of sea-level anomalies in a multi-model prediction framework" (JGR-Ocn)

Altimetry assimilation improves initial conditions and seasonal forecast skill

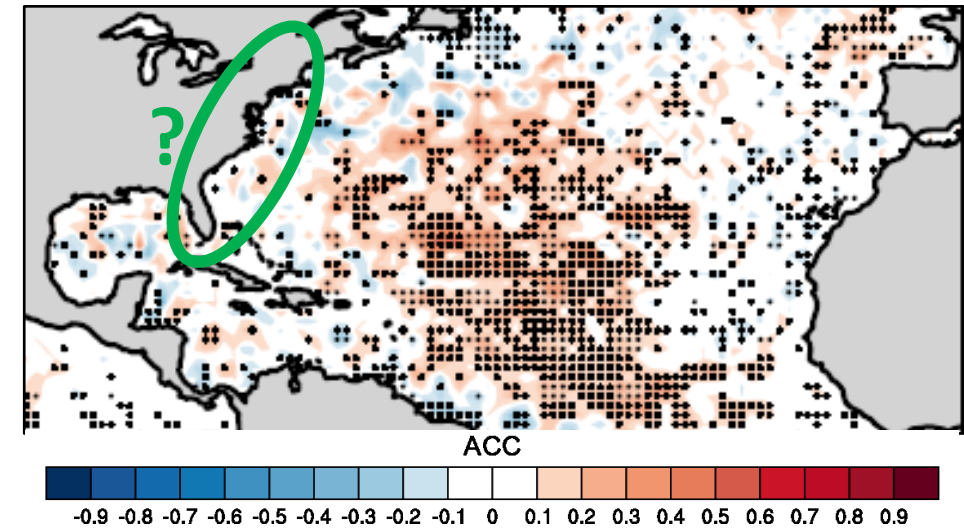
Comparing altimetry observations with forecasts of SSH
SEAS5-Control minus SEAS5-Experiment (0.05)



SEAS5-Control minus SEAS5-Experiment (0.03)



Lead-0 month difference

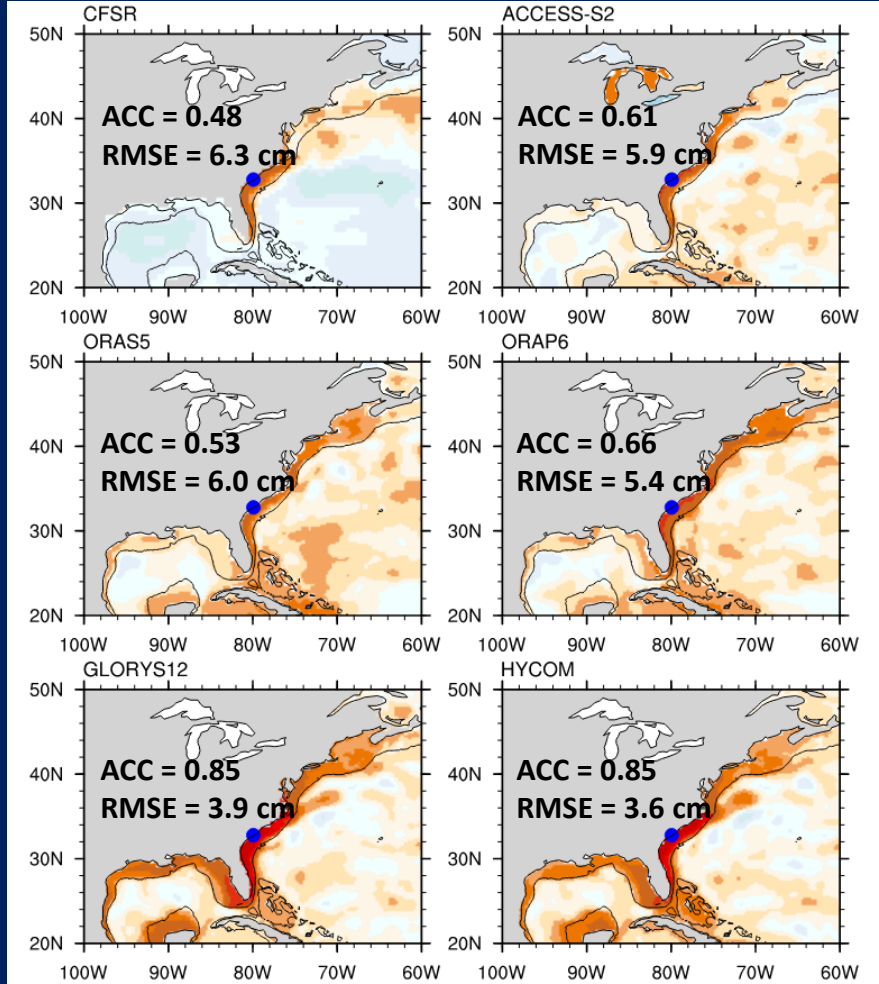
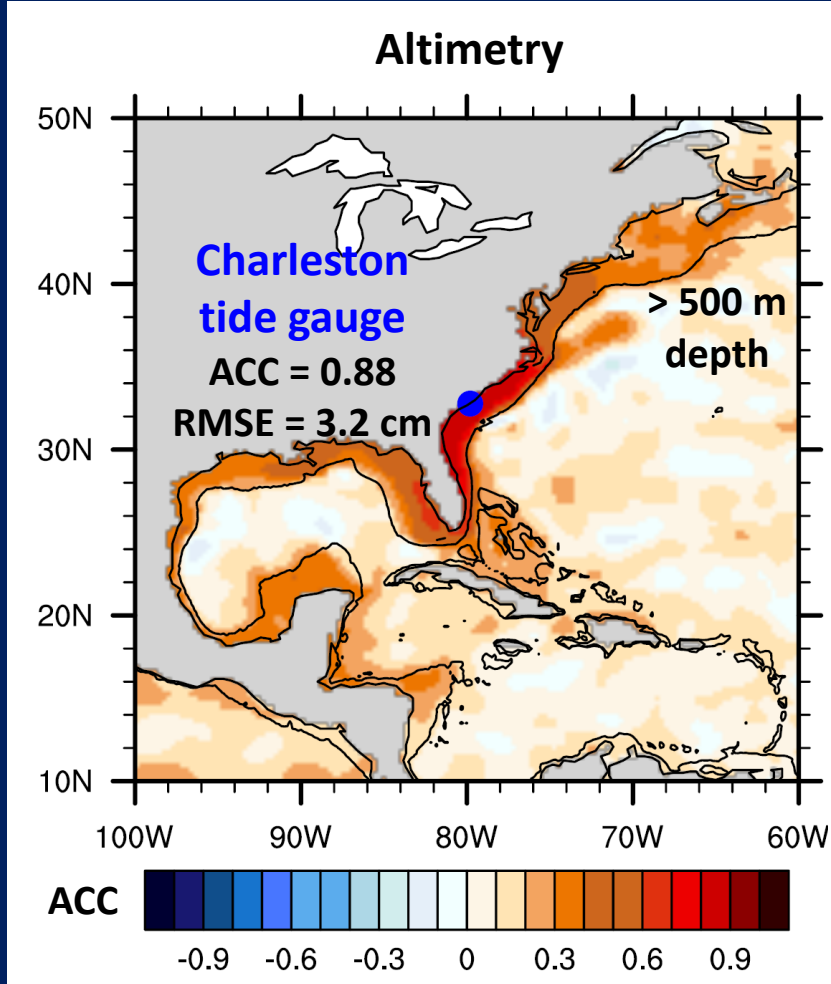


Collaboration:
ECMWF (SEAS5-Experiment) with
altimetry assimilation turned off

Widlansky et al. 2023

*“Quantifying the benefits of altimetry assimilation
in seasonal forecasts of the upper ocean” (JGR-Ocn)*

Some reanalyses resolve monthly sea level variability almost as well as altimetry, including near the coast



*Linear trends removed

Analysis by Xue Feng (manuscript in prep)


AI helping use ocean observations



How do I convert sea levels from Honolulu to be with respect to MSL? + Tidal datums metadata



To convert the sea level data from UHSLC to be with respect to Mean Sea Level (MSL), you can subtract the MSL value from each sea level measurement.

 Here's the code to do this using Python:

```
```python
sea_level_data = [1.258, 1.342, 1.176, 1.409, 1.251] # example sea level data

Convert to MSL
MSL = 1.412 # Mean Sea Level value

sealevel_MSL = [x - MSL for x in sea_level_data]
print(sealevel_MSL)
```
```

1) **Question** asked (front end)

2) **Metadata** appended (back end)

3) Large-Language Model **creates** (OpenAI)

```
prompt = question + ' ' + json.dumps(metadata)

openai_res = openai.ChatCompletion.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": prompt}
    ]
)
```