Ray Tracing Ocean Surface Waves with Rust

Bryce Irving[1], Guilherme Castelão[2], Bia Villas Bôas[1]

1 - Colorado School of Mines

2 - Scripps Institution of Oceanography, UC San Diego

Motivation

Ocean surface gravity waves play a major role in the exchange of momentum, heat, energy, and gases between the **ocean** and the **atmosphere** [Villas Bôas and Pizzo, 2021].

- **Wave propagation** is impacted by changes in ocean currents and bathymetry (depth)
- **Full wave models** are **computationally expensive** and complex to interpret
- **Ray-tracing** is a powerful technique used in computer graphics and scientific simulations to efficiently **model the propagation of waves**, providing insight into the spatial variability of wave properties such as amplitude and direction

Theory

The ray equations for ocean surface gravity waves, neglecting the effects of currents, give the following set of **ODEs** that can be **solved numerically** [Phillips, 1977]:

$$1.\ \frac{dx}{dt} = c_{g_x} \qquad 3.\ \frac{dk_x}{dt} = -\frac{\partial \sigma}{\partial x}$$

$$2.\ \frac{dy}{dt} = c_{g_y} \qquad 4.\ \frac{dk_y}{dt} = -\frac{\partial \sigma}{\partial y}$$

Where, *(x, y)* is the position, *t* is the time, (c_{gx}, c_{gy}) is the group velocity, and (k_x, k_y) is the wavenumber. The frequency σ is related to the wavenumber *k* by the linear dispersion relationship:

$$\sigma = gk[\tanh(kH)]^{1/2}$$

where *H* is the water depth. Additionally the group velocity is defined as:

$$c_{g_i} = \frac{\partial \sigma}{\partial k_i}$$

Objectives

Develop an **open-source Rust package** that solves the **ray tracing** equations for **ocean** surface **waves** in a computationally efficient and robust way:

- Simulate **deep and shallow water** at constant depth
- Trace wave through varying **bathymetry**
- Trace wave accounting for **surface current** effects
- Visualize the results
- **Parallelize** the ray tracing

Rust provides, **high-performance**, **robustness**, and **reliability,** making it seamless to run **large ensembles** of ray tracing simulations.

Example idealized test case:

- Circular island
- Constant bathymetry slope around the island

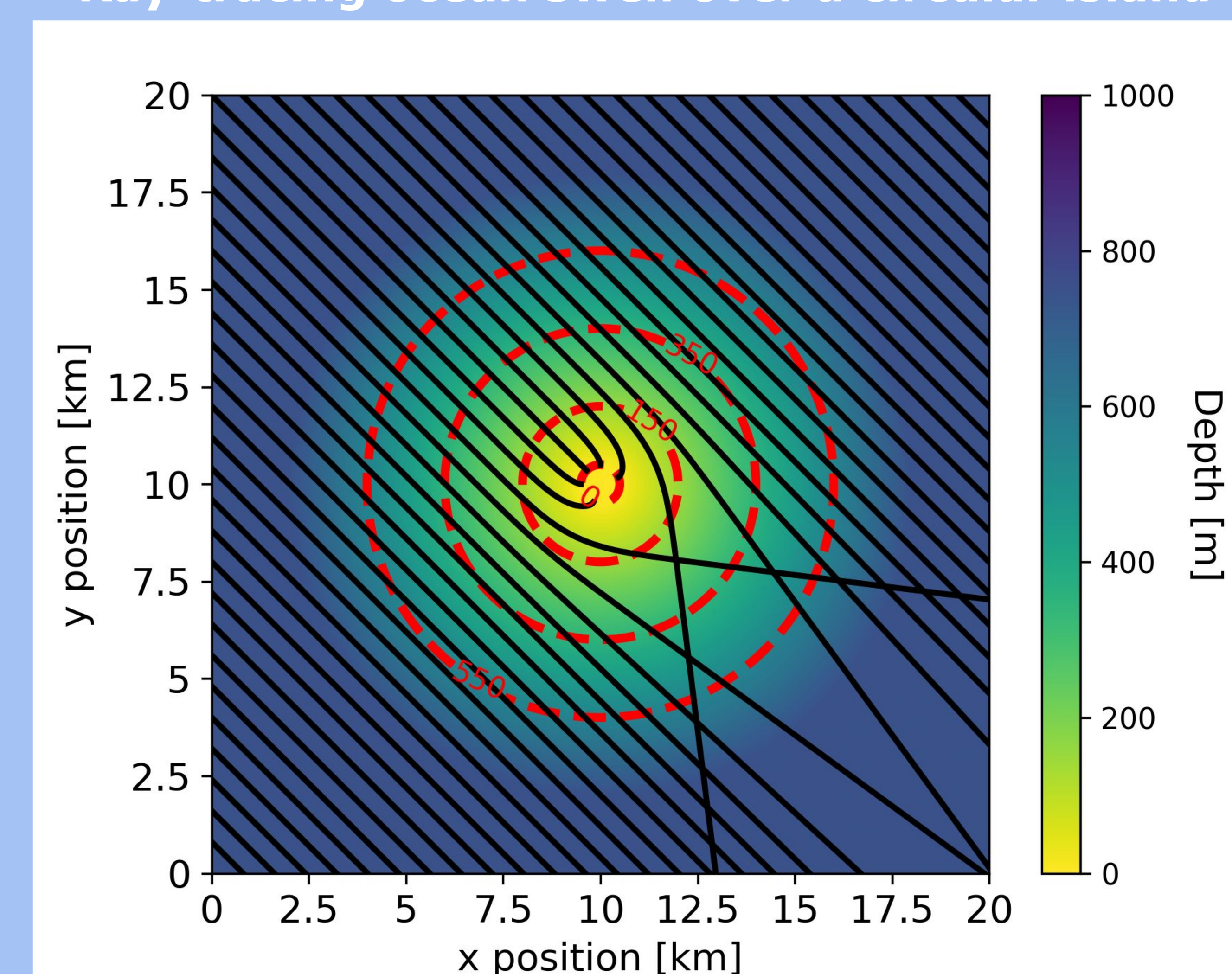
Initial conditions:

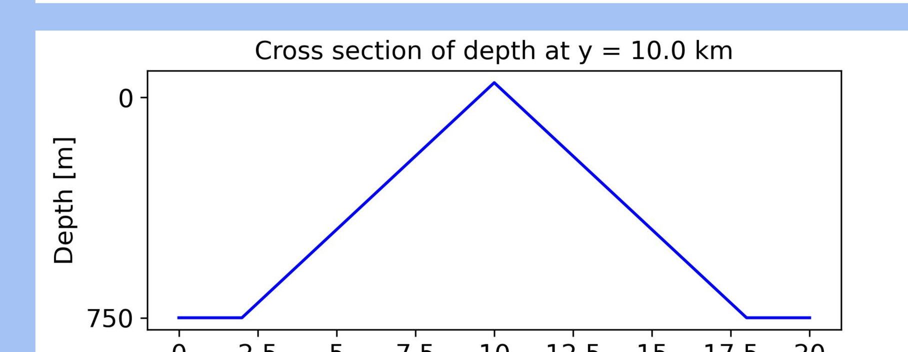
- Wavenumber k_0 = 0.0125 1/m, (corresponding to wavelengths of ~500 m) entering the domain from θ = 45º

Results:

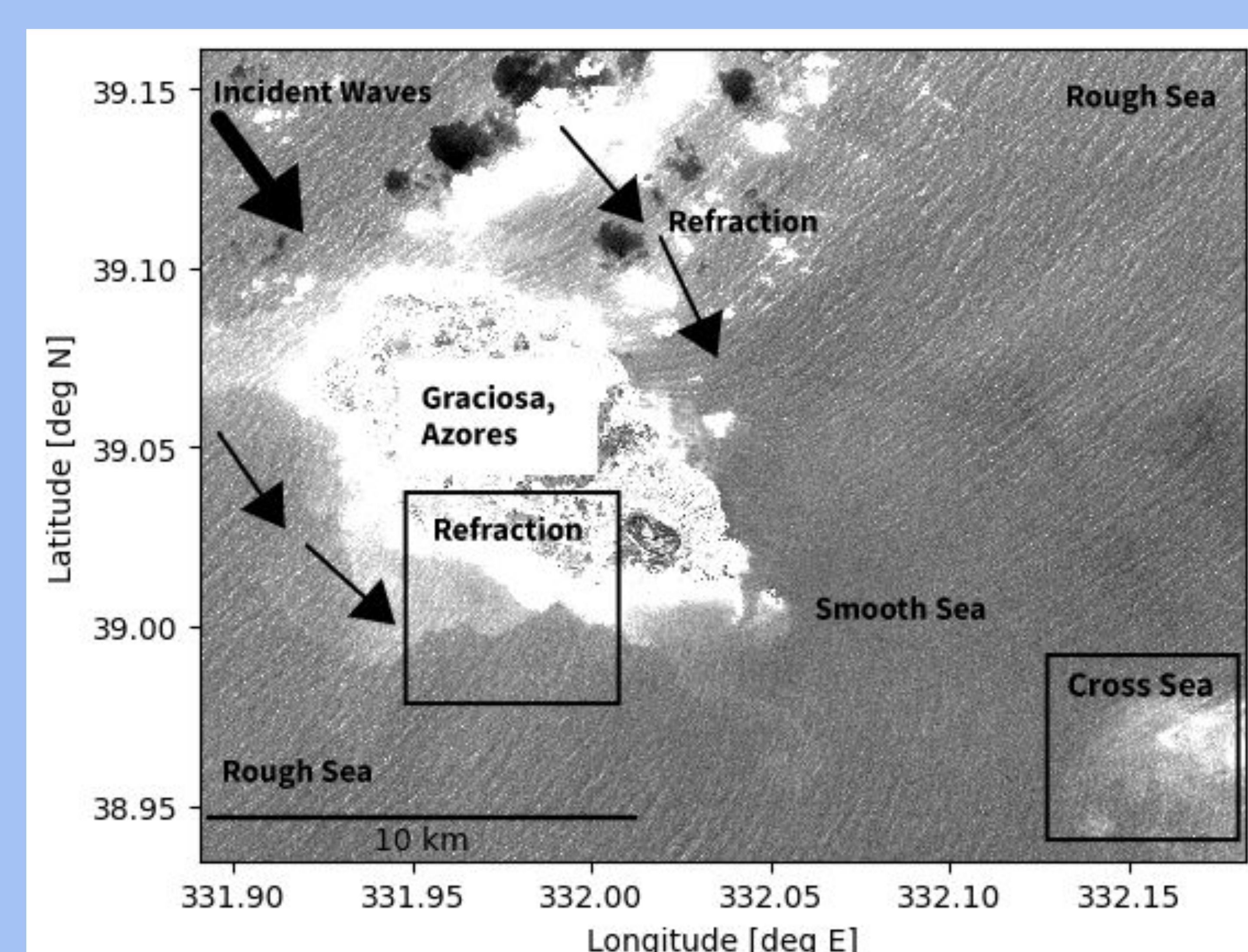
- **Waves refract**, or bend, around island due to **changes** in water **depth**
- Island blocking and refraction creates a shadow zone on southeast side of the island
- Results compare well with observations (below)

Ray-tracing ocean swell over a circular island





Cross section of bathymetry at y = 10 km.



Sentinel-2 satellite image of the sea surface roughness around Graciosa Island, Azores for similar incoming swell conditions:

- Refraction around the island
- Shadow zone on south-east of island
- Close-ups show cross-seas and refraction

Methods

We used an **efficient** language, **accurate** algorithms, and continuous **testing:**

- **Rust** is a low level language that provides:
  - High performance
  - Memory Safety
  - Convenient testing
  - Cross-compilation support for Linux, Mac, and Windows
- **RK4** and **bilinear interpolation** reduce error from:
  - Integrating ray equations
  - Interpolating the depth
- There is continuous **testing** using:
  - CI/CD using GitHub Actions on different OS
  - Simple unit testing in Rust
  - Idealized test cases

Future Improvements

We will improve realistic **accuracy** and **usability** in the **future**:

- Implement method to generate initial waves improving the user interface.
- Include the effects of **surface currents.**
- Expand coordinate systems to include spherical coordinates (**latitude and longitude)**.
- **User friendly** use with **python** foreign function interface bindings.

Conclusion

Rust has proved **fast, accurate**, and **robust**, thus better for numerical calculations:

- Rust crate performed well in **ideal conditions**.
- Our **future improvements** will increase utility and allow ray tracing with actual **bathymetry files**.
- The **foreign function interface** will allow time-intensive calculations to be performed in Rust, while the user still writes their **main code in Python**.
- We plan on releasing the ray tracing crate as an **open-source software package** to contribute to the ocean sciences community.

References

Villas Bôas, A. B. and N. Pizzo (2021). The geometry, kinematics, and dynamics of the two-way coupling between wind, waves, and currents. pp. 18–26. US CLIVAR.

Phillips, O. M. (1977). The Dynamics of the Upper Ocean, 2nd Edition. Cambridge University Press,. 336 pp.

Acknowledgements

Funding for this work was provided by the Mines Summer Undergraduate Research Fellowship and NASA grant 80NSSC23K0979.

We thank Gwendal Marechal for his help in providing the Sentinel-2 images, python scripts, and annotation advice.