UCONN
UNIVERSITY OF CONNECTICUT

MARN5895-N60
Research Computing in Marine Sciences
Department of Marine Sciences
University of Connecticut

Fall 2021 Syllabus
v1.0, August 18, 2021

Excluding materials for purchase, this syllabus may change in the course of the semester. The most up-to-date version will be available on HuskyCT.

Course information and intructor

Course title: Research Computing in Marine Sciences
Credits: 3
Modality: Hybrid/synchronous (in-person, but prepared to go online if needed)
Prerequisites: None
Course times and location: Tuesdays and Thursdays, 8:00-9:20.

Instructor: Cesar B Rocha, Assistant Professor of Marine Sciences. My pronouns are he/him/his, and I prefer that you call me by my first name (pronounced CEH-zar, unlike the salad; the *e* in the first syllable is vocalized as the *e* in *bet* and *met*). My office is Room 188 in the Lowell P. Weicker Jr. building (aka Marine Sciences building), and my email is cesar.rocha@uconn.edu.

Office hours: TBD.

Course materials

Course Web site: I will keep the most up-to-date class schedule, including the assignment and suggested reading for each week, on a Github repositoty. Class materials (scripts, Jupyter Notebooks, cheat sheets) will also be shared on this repository.

Required hardware: Students need a working laptop computer they can bring to class. I recommend a computer with at least 8 GB of RAM and 20 GB of available disk space and running Mac OS or Windows or Ubuntu Linux. Please, let me know if you do not have a laptop that meets these requirements.

We will work remotely on the Storrs HPC (High-Performance Computing) facility. To register for a free account, students should visit `hpc.uconn.edu`, click on the *Storrs Fa-*

cility drop-down menu, and select *Get an Account*. When working off-campus, students will need the UConn VPN for accessing Storrs HPC.

Required software: This course prioritizes open software freely available online. I will set up a Web site with a list of basic software requirements and detailed instructions for software installation. Before the start of the semester, we will organize an optional hackathon for installing software such as git for version control, Visual Studio Code for a nice text editor and terminal, Miniconda for the scientific Python stack. I will also use the in-class tutorials to share tips and tricks for installing, troubleshooting, and updating software.

Students may occasionally need University-licensed proprietary software (Excel, Matlab, Mathematica), which are available through UConn Information and Technology Services.

Optional materials: Below is a list of optional resources to support your learning. They are all freely available online.

- The Python Data Analysis Handbook by Jake VanderPlas.
- Whirlwind Tour of Python by Jake VanderPlas
- Think Python by Allen B. Downey.
- The Official Python Tutorial.
- The Software Carpentry Lessons.
- The Scientific Python Lectures.
- The Python For Geosciences Lectures.
- PEP8: Style Guide for Python Code.

Course description

This course will introduce students to modern computer software, programming tools, and best practices for reproducible research, focusing on data-driven applications in Marine Sciences. Students will learn how to use the terminal shell, work remotely on supercomputers, version control their code, collaborate using git/Github, analyze and visualize data in Python. Through practical assignments and participatory lessons peppered with tips and tricks, students will develop the self-confidence to tackle problems computationally, unlocking a potential to harness ever-larger datasets in their research and beyond. Students will also employ their newfound computing skills to make their findings accessible and reproducible, ensuring that anyone can check their work and build on it.

Lessons and assignments will use oceanographic datasets, including observations collected by in-situ and remote platforms (Argo floats, surface drifters, Saildrones, satellites, coastal radars) as well as model reanalysis products (CMIP outputs, ECCO, ORAS). Based on my background, this list is biased towards physical oceanography. But I'm interested in using non-PO datasets for class demos and assignments. *Let me know which datasets you would like to learn how to access and work with.*

This course's pedagogy borrows heavily from Greg Wilson's decades of experience with Software Carpentry, culminating in his book Teaching Tech Together.

Course objectives

This course's overarching objective is to **empower students to use modern computer software to solve problems**, with an emphasis on data-driven applications in Marine Sciences. By the end of the semester, students will be able to

1. **Navigate** the computer file system and **automate** tasks using the Unix shell.
2. **Work** remotely on Storrs HPC and other High-Performance Computing systems.
3. **Read**, **analyze**, and **visualize** large datasets using Python/Jupyter Notebooks.
4. **Understand** the advantages and limitations of different programming languages.
5. **Access** and efficiently **download** oceanographic datasets that live in the cloud.
6. **Version-control** and **backup** computer code and **collaborate** using git/Github.
7. **Employ** computational workflows for practicing reproducible research.

Course structure and schedule

We will meet twice a week. In the first meeting of the week, I will teach a tutorial-style lesson on the week's topic. This is an **active-learning course**. Students are required to follow along the tutorials using their laptops, solve in-class exercises, and help their classmates. In the second meeting of the week, we will **flip the classroom**, organizing hackathon-style sessions where students will work on assignments, interact with other students and me, and present their work in progress. Students will work in pairs or small groups on these assignments and submit a short report due the following week.

Here's a rough weekly program:

Week 1
Lesson: *Introduction to the class, the Unix shell, and Storrs HPC.*
Assignment: *Installing miniconda on Storrs HPC.*

Week 2
Lesson: *More on the Unix shell.*
Assignment: *Moving and renaming a large number of files.*

Week 3
Lesson: *Python basics and the Jupyter Notebook.*
Assignment: *Visualizing Temperature, Salinity, and CO_2 data collected by Saildrones.*

Week 4
Lesson: *Version control with Git and Github.*
Assignment: *Improving the class Web site through a pull request on Github.*

Week 5
Lesson: *The Python scientific stack.*
Assignment: *Analyzing and plotting data from Argo floats.*

Week 6
Lesson: *Pandas for tabular data.*
Assignment: *Estimating the sea-level trend in Connecticut from coastal tide gauges.*

Week 7
Lesson: *Xarray for multidimensional arrays.*
Assignment: *Estimating the global sea-level trend from 30 years of satellite altimetry.*

Week 8
Lesson: *More on Xarray and CF-compliant netCDF files.*
Assignment: *Estimating the ocean heat content and its trend from Argo products.*

Week 9
Lesson: *Cartopy for making maps.*
Assignment: *Making composite plots with several datasets in the Mid-Atlantic Bight (coastal HF radars, satellite SST, Chlorophyll-a).*

Week 10
Lesson: *Cloud computing.*
Assignment: *Analyzing satellite altimetry data with Jupyter Notebooks on Amazon Web Services and Google Cloud.*

Week 11:
Lesson: *Parallel computing with Dask and Pangeo.*
Assignment: *Estimating global warming in a climate model ensemble.*

Week 12
Lesson: *Reproducible workflows (testing, documentation, publication, archiving).*
Assignment: *Documenting and publishing your code on Zenodo.*

Week 13
Lesson: *Advantages and limitations of other languages (R, Matlab, Julia).*

Assignment: *Running R or Julia on Jupyter Notebooks.*

Week 14
Quiet week. Students will work on finalizing their projects; I will be available in the classroom for help.

Week 15
Final presentations.

Course requirements and grading policy

Assessment
Students are required to *actively participate in the class* (20%), complete *weekly assignments* (30%), and complete and present a *final project* (50%). The table on the right shows the grading scale for the course.

Score rage	Letter grade
≥93.0	A
90.0–92.9	A-
87.0–89.9	B_+
83.0–86.9	B
80.0–82.9	B-
77.0–79.9	C_+
73.0–76.9	C
70.0–72.9	C-
67.0–66.9	D_+
63.0–66.9	D
60.0–62.9	D-
<60	F

Active participation involves following the tutorials on your computer, engaging in discussions, helping your colleagues, and presenting your in-progress work. For the assignments and final project, students will work in pairs or small groups. The assignments are small computational projects related to the week's topic. The final project should involve obtaining, processing, and visualizing a large dataset (e.g., satellite altimetry, biogeochemical Argo, coastal high-frequency radar, CMIP outputs) using the tools learned in the course. In consultation with me, student pairs or small groups should choose their dataset by week 7. The final project must be well documented and shared on a Github repository. The other students in the class and I should be able to clone your repository and reproduce your work as you present the final project in Week 15.

Due dates and late policy
All due dates will be identified on the course Web site. Deadlines are 11:59 pm Eastern Standard Time. I may change dates as the semester progresses. All changes will be communicated at least one week prior to the new due date.

If you have a good reason (caregiving, illness, hardships), please do not hesitate to ask for an extension *prior to the deadline*. No late assignments will be accepted unless previously discussed with me. Students with learning disabilities are eligible for extended due dates and other accommodations through the Center for Students with Disabilities (see Students With Disabilities below).

Feedback
I will strive to provide feedback on your assignments within a week of the due date. I also commit to sharing mid-semester partial grades to help you plan for the second half of the term.

How to ~~succeed~~ thrive in this class

You will succeed in this class—if you have a desire to learn and are willing to roll up your sleeves. Below are a few tips from yours truly (who sat where you sit not so long ago) to help you maximize your learning.

- **Engage** in the class activities

  Engaging with me and your classmates is critical to learning the material and keeping yourself motivated. When I ask questions, answer them. When you have questions, do not hesitate to speak up. If something is unclear to you, the chances are that it's also unclear to your classmates. Discuss the material with your classmates and comment on their presentations. Take the initiative to present your work in progress. Let me know if the structure and pace of the class are working for you. And please call out my mistakes.

- **Do NOT multi-task**

  Computers are particularly good at executing several threads of tasks in parallel. In Week 11, we will learn how to leverage tens to hundreds of cores to increase computing performance. But we are not computers and our brains are not multi-threaded. Because you will be required to work on your computer, which invites you to multi-task, you must be mindful that multitasking decreases human productivity and generates a constant humming of anxiety. The work we will be doing in this class requires undivided concentration. Do not check your email inbox or social media in class or when working on your assignments. Unless you are expecting an urgent text or call, put your phone on airplane mode and turn off notifications. Stay focused on the task at hand.

- **Tinker** with code developed in class

  A few hours or a day after each class, review the material covered in the tutorial. That includes re-running the codes developed and used in class. To make sure you understand how a code works, tinker with it. Change parameters and conditions. Generalize specific functions. Apply the code to different datasets. Make different plots.

- **Help** your colleagues

  Students in this class come with diverse backgrounds and levels of programming experience. Some students may stumble on a particular code during in-class activities or may find completing an assignment challenging. To avoid frustration getting the most of your colleagues, you should help them out. Helping and teaching classmates and friends is a powerful approach to making sure you understand

the material. Of course, when you are helping someone, make sure you are explaining things to them, not doing their work. One of my goals with this class is to show all students that programming can be fun and collaborative, and I count on you to help me realize that goal.

- **Incorporate** the tools learned in class into your daily workflow

  The best way to make the material stick is applying the concepts and tools learned in class to your daily work. Use the terminal shell more often to navigate the file system and automate tasks. Rewrite an old Matlab or R script in Python. Load and manipulate an Excel spreadsheet using Pandas. Use Python to analyze and plot your research data. Use Github and git to version-control and backup your code. Use Github to collaborate with colleagues in other classes and your lab.

Students Responsibilities and Resources

UConn students are held to certain standards and academic policies. Please, review the following policies:

- Absences from Final Examinations
- Class Attendance
- Credit Hour
- People with Disabilities, Policy Statement
- Discrimination, Harassment and Related Interpersonal Violence, Policy Against
- The Student Code
- Academic Misconduct Procedures for Instructors
- Scholarly Integrity in Graduate and Post-Doctoral Education and Research

Students with Disabilities

The University of Connecticut is committed to protecting the rights of individuals with disabilities and assuring that the learning environment is accessible. If you anticipate or experience physical or academic barriers based on disability or pregnancy, please let me know immediately so that we can discuss options. Students who require accommodations should contact the Center for Students with Disabilities, Wilbur Cross Building Room 204, (860) 486-2020 or csd.uconn.edu.

Software Requirements

Most of our work will happen on StorrsHPC. To access StorrsHPC smoothly, you should install Microsoft's enhanced text editor Visual Studio Code if you use Windows. Visual Studio Code has a Unix-like terminal built in. If you use MacOS or Linux, you only need a terminal running bash, but I recommend that you install Visual Studio Code too (it has a neat text editor). Regardless of your operating system, you will also need a modern internet browser (preferably Google Chrome or Mozilla Firefox).

Minimum Technical Skills

This is a course on research computing, so you will need to be comfortable working on computers. To follow along the tutorials and complete the assignments, you should be able to install and open different software on your computer (Visual Studio Code, internet browser), copy and paste text across different software (e.g., from browser to Visual Studio Code), toggle between different tabs in the browser and terminal. For your assignment reports, you should be also able to generate PDF files.

If you are unsure whether your technical skills are sufficient to succeed in this class, please contact me.

Help

If you have trouble accessing the course Web site, please contact me. If you have difficulty accessing StorrsHPC, contact their support at hpc.uconn.edu/storrs/support. If you have issues with HuskyCT or VPN, contact HuskyTech during regular business hours. For further assistance, contact Course Support, available 24/7.

Course Evaluation

Students are strongly encouraged to provide feedback on their course experience and instruction through the Student Evaluation of Teaching (SET), administered by the Office of Institutional Research and Effectiveness (OIRE). The SET is used for both formative (self-improvement) and summative (evaluation) purposes. UConn commits to supporting and enhancing teaching effectiveness and student learning using a variety of methods. In addition to SETs, two tenured professors will seat in one lecture to observe the instructor. The instructor will also conduct informal mid-semester formative surveys via HuskyCT.