Introduction to Spatial Computational Programming 2019 v1.

Contributors/Reviewers:

@koyo_jakanees @Pondi_pb @Nombzzz @wanjohi-kibuhi @vic_ngeno @ochwadalinda

This course is to serve as an introductory level to spatial computation workflow and modern day geospatial engineering and product(software) deployments.

It assumes basic knowledge of basic spatial and geoprocessing operations using common Geospatial information System Software products like *ArcGIS, QGIS, MapinfoTab, Autodesk ,Whitebox tools, e.t.c*. With the recent popularity of computer science domains of artificial intelligence there has also been an exponential rise in the need for High perfomance Computation but if you have been in the geospatial related field this need is not new to you maybe if you've been dealing with gigabytes of data either raster or vector that required a lot of processing power and optimization of processes.

They are quite a number of programming languages used in the geospatial domain from ***python , javascript,R, C/C++,spark, scala, Rust to Fotran*** . Each of the languages have their pro`s and cons from speed to verbosity or even meer popularity; but the type of mentality one should have is that "whatever can be done in one language can be done in any other programming language" and there are existing inter-language wrapper to access functionality between one another. For example GDAL(Geospatial data abstraction Library) one of the most used software for different spatial file formats I/O and conversion is written in C/C++ but the functionality can be accesed in R using **rgdal** or python using **gdal of ogr** modules from osgeo.

Since this module is not focused on any specific subdomain of geospatial industry, the language used will be python and reletated modules for the spatial operationsin the free and opensource scope therefore proprietory modules will not be subject to this scope.

- Spatial visualization: R, javascript, python, etc.
- Geoprocessing and/or Geostatistics: python, R,Spark, C++, C#,octave etc
- Spatial web development: Javascript, python, R, C++,C#,Scala.etc
- Raster Processing: C++, python, R, Javascript,C++, Matlab, octave etc

Above list is not definitive but just to show how the same tools can be used to achieve similar tasks, thus it's adviceable to pick the tool(your swiss knife) that you can obtain the end results without a steep learning curve and is comfortable with

This material does not serve as a complete guide to spatial analytics and geoprocessing but as an appetizer to solving basic spatial problems by designing simple algorithms and re-use of already exiting ones without "re-inventing the wheel" and keeping it DRY(don't repeat yourself technique) using python programming language..

The main goal of is to provide a programmer/computer science set of thinking to the students participating.

Objectives of the Presentation

Pre-course

I. **Git for Spatial Scientists:**

Overview of version control system (git) and online repository hosting services such as github/gitlab/Bitbucket repositories

```
Software used to keep track of your files and backing up online. This proves vital
in collaboratory research and programming; also at a personal level to manage your
files

One should strive to grasp the use of 'intialising a repo, cloning, pulling or
pushing to a remote repo merge, committing locally'. Learn just enough to allow
you to comfortably work through the basics.
```

> *Here is a basic overview of how Git works:*
>
> - create or initialize your local repo/dir(barebone folders with) with `git init` or
> - Create an online/remote "repository" (project) with a git hosting tool (like Bitbucket or github)
> - Copy (or clone) the repository to your local
> - Add remote alias to your remote repo
> - Add a file to your local repo and "commit" (save) the changes
> - "Push" your changes to your master branch
> - Make a change to your file with a git hosting tool and commit
> - "Pull" the changes to your local machine
> - Create a "branch" (version), make a change, commit the change
> - Open a "pull request" (propose changes to the master branch)
> - "Merge" your branch to the master branch

Links to Resources

- Download git choose the one for your Operating system
- Vesrion Control with git **Recommended** Approximately 3~min introductory walthrough goal oriented i.e collaboratoring with git from software capentries
- Github intro
- Bitbucket*worth checking out after Software capentry:* short graphical tutorial

- Pro Git book comprehensive guide of git recommended on their official website

> If comfortable enough and has appetite for more, check Geogig which is *an open source tool that draws inspiration from Git, but adapts its core concepts to handle distributed versioning of geospatial data.* Inspired by git but domain specific for tracking spatial data changes from shapefiles, spatialite, osm, postgis e.t.c Check out the offical workshop walk through to explore at your own pace.

Note:

When working on any project from scientific research or software developement or class work or content development besides being able to keep track of changes that you make in each step using version control system it's also **Important:** to document your workflows, reports, hack arounds in cases of a bug or error for instance this document. such documents are written in mark up or markdown languages such *html ~ emmently used in creating websites, Latex used frequently in academic research and report writting, markdown like this one; github flavoured markdown*

I suggest you take roughly 5 minutes to Github Flavoured markedown guides to familiarize and try out different tags.

II. **Introduction to terminal/shell**

Brief discussion on using 'nix based shell/terminal/command prompt

```
Text based set of programs that ships default with your pc/laptop that proves
helpful in writing automation scripts navigating through directories/folder,
files. Before the rise of the Graphical User Interfaces laptops were text based
where instead the beautiful User interfaces back in the 70's.

Due to the current progress made in cloud computing, familiarity in the *nix based
commands gives the student an easy time through prefered cloud service e.g google
GCE, amazon AWS etc
```

- Software Capentry Unix Shell less than 5 minute tutorial to show case a typical unix shell operations in data intense environments
- Surrey Unix Introduction

III. **Introduction to computational thinking with python**

Overview of programming thinking in solving obvious and ubiquitious real world problems using computer science principles and python programming

---

Essential Concepts

> *An excerpt from Computational Models Dr. Jose M. Garrido 2016*

> > A computational model is a computer implementation of the solution to a (scientific) problem for which a mathematical representation has been formulated. Developing a computational model includes formulating the mathematical model and implementing

> it by applying Computer Science concepts, principles and methods such as: **abstraction, decomposition, specification, conceptual modeling, programming, data structures, algorithms, design structures, verification, and validation.** Computational modeling is the foundation of Computational Science.

> Abstraction is recognized as a fundamental and essential principle in problem solving and software development. Computational Science integrates concepts and principles from applied mathematics and computer science and applies them to the various scientific and engineering disciplines. Computational science is an emerging area (or discipline) that involves developing computational models applied in various areas of science and engineering to solve large-scale (and complex) scientific problems. Computational Thinking is required for any problem that involves calculations. This involves deciding: what calculations use what values, and the order in which the calculations must occur. It usually requires multi-disciplinary and team work. Computational thinking is the ability to describe the requirements of a problem, design a mathematical solution to the problem, and implement the solution in a computer. Large and complex computational models usually require high performance computing resources to execute and the models are used to study the behavior of large and complex real systems or processes.

Understanding the concepts of solving problems systematically and designing the solutions to this problems(recursively and algorithmically) using code. Derive and write simple programs to compute cuberoot, pallidrome strings etc without using in built modules or functions.

Emphasis of this section is not to memorise the syntax of python language but how to design and descibe each and every single step of the solution in simple plain english, then worry about the code implementation afterwards.

At the end the student should be familiar with python syntax style, defining functions, variable creation and assignment, different data type and the declaration, control flow and branching.

introduction to external modules and libraries: How they are installed and can be accessed e.g. Numpy, Bokeh, fiona,shapely, rasterio, pysal, pandas, geopandas.

> **Familiarity with Numpy and Pandas** python modules used in numeric computation and read data tables is desired but not reguired python Dev stack open source modules. awesome Geospatial