# 

# 

# **FUNWAVE-TVD SHORT COURSE TUTORIAL**

# **Sept 18, 2022**

# 

# 

# 

# 

**Prerequisites for the FUNWAVE-TVD short course**

* Your laptop is able to use Wi-Fi to access the internet
* You can use SSH Secure Shell Client (like: Putty or Cygwin on Windows)
* **If you are using a Mac OS X or a Linux laptop**, you will need a working suite of compilers and Message Passing Interface (MPI) wrappers to build a parallel version of FUNWAVE-TVD.   
    
  **(A)** For **macOS** it is recommended (if you already don’t have them) to install GNU compilers (gcc, g++, gfortran) via Homebrew. If you don’t have Homebrew installed type the following in your terminal:
  + **cd ~**
  + **mkdir homebrew**
  + **curl -L https://github.com/Homebrew/brew/tarball/master**
  + **tar xz --strip 1 -C homebrew**

If you didn’t previously have these compilers on your system (say via XCode) then the   
 gcc, g++, gfortran would be placed under **/usr/local/bin/** directory, otherwise they   
 could be placed under **/usr/local/Cellar/gcc/9.1.0/bin/**. Then you will need   
 to install GNU suite of compilers using `**brew**`, where the current versions are 9.1.0   
 (**\*Important:** you want to use all suite components [gcc, g++, gfortran] of the same   
 version to avoid possible issues/compiler conflicts later on.)

* + **cd ~**
  + **brew install gcc**
  + **brew install gfortran**

Finally, you will need to install MPI compiler wrappers. Note that if you had a trace of

previous compilers, you may need to explicitly tell Homebrew during installation to   
 point to/use the newly installed compilers via a FULL PATH in you terminal (e.g.,   
 **export HOMEBREW\_FC=/usr/local/Cellar/gcc/9.1.0/bin/gfortran &&   
 export HOMEBREW\_CC=/usr/local/Cellar/gcc/9.1.0/bin/gcc &&   
 export HOMEBREW\_CXX=/usr/local/Cellar/gcc/9.1.0/bin/g++)** prior to   
 executing MPI installation command [recommend using MPICH], as in

* + **cd ~**
  + **brew install mpich**

**(B)** For **Linux**, you can use either your system-wide installation commands in your   
 terminal/command line (**if you have admin/sudo privileges**). For Ubuntu, Debian,   
 Mint, and other Debian-based distributions type the following in your terminal   
 [provided you already don’t have a working suite of these: check via `**which gcc**`,   
 `**which g++**`, and `**which gfortran**` first]

* + **sudo apt install gcc**
  + **sudo apt install gfortran**
  + **sudo apt install mpich**

For Red Hat, Fedora, or CentOS Linux kernels, you would replace `**apt**` in the three   
 commands above with `**yum**`.

Alternatively, you can also use Homebrew to install the required packages. See here   
 for more details: <https://docs.brew.sh/Homebrew-on-Linux> . It should be noted that   
 Homebrew only does local installation (no system-wide `sudo` commands for   
 installation are allowed).

* **If your laptop uses Windows OS lower than Windows 10**, it is recommended that you download the latest version of cygwin (<http://www.cygwin.com>). Cygwin is a bash shell/unix emulation program and contains many of the tools such as tar, gzip/gunzip, and cpp, which will be useful for installation and compilation of FUNWAVE-TVD. In addition, If you don’t have access to a High Performance Computing (HPC) machine at your home location, it is recommended you can use the **Amazon AWS Cloud Computing**; setup and free use is described at the end of this tutorial for this workshop. Note that the instructions on how to install FUNWAVE-TVD on **Windows10** can be found at: <https://fengyanshi.github.io/build/html/windows.html>
* You have a post-processing toolbox on your machine (laptop). Both MATLAB and Python post-processing script are provided with most practice examples. Many participants already use Matlab, **but if you do not, it is recommended that you install an Anaconda Python package (information below) for any platform.**

**Downloading and Installing a Python Package** (Anaconda)

Arguably one of the best and most comprehensive **FREE** packages for the Python language, along with most tools and modules (e.g., NumPy, Matplotlib, etc.) is distributed by the Continuum Analytics under the Anaconda package. It is available for Linux, Mac OS X, and Windows machines. You do **NOT** need administrator privileges to install the Anaconda package, you can do so as a standard user on all three platforms listed above.

For the Anaconda package go to: <https://www.anaconda.com/distribution/>

Pick the **appropriate platform** (Linux, Mac OS X, Windows) by clicking the correct tab and get the Anaconda distribution that comes with **Python 3.7** (not 2.7). You can either download the **Graphical Installer** (recommended), of if you are comfortable with the terminal in the Linux/Mac OS X environment, you can also download it through the command line.

For the complete list of packages/modules included in the Anaconda Python distribution see: <https://docs.continuum.io/anaconda/packages/pkg-docs>

**Backup Plan:** if you don’t have HPC access or you laptop is still not cooperating,  **use Amazon AWS Cloud Computing**

* Amazon AWS EC2 provides 12-month free tier, 760 hours/month
* If you don’t have an HPC cluster access, you can build your cluster in EC2
* A simple example can be found in **Appendix** of this tutorial

**Training Session # 1**

**Topics**

* FUNWAVE-TVD and Parallel Computing (MPI) - Documentation Wiki   
    
  Wiki ⇒ <https://fengyanshi.github.io/build/html/index.html>
* Where do I get the code? - Version Control (Github)

Full Repository ⇒ <https://github.com/fengyanshi/FUNWAVE-TVD>  
  
Latest Code Release (July 2019) is version 3.4 ⇒ <https://github.com/fengyanshi/FUNWAVE-TVD/releases/tag/Version_3.4>

**Code and examples for this short course (recommended)**

<https://github.com/fengyanshi/Tsunami_workshop>

* How to build (compile/link) and install FUNWAVE-TVD on different machines for parallel computation?  
    
  <https://fengyanshi.github.io/build/html/setup.html#compile-and-setup>

**Practice**

1. Log into ⇒ your HPC [if working on your laptop, skip to step 2 below]

> **ssh** [**your\_user\_id@hpc.udel.edu**](mailto:your_user_id@hpc.udel.edu) **(example)**

... input your password

> **mkdir your\_funwave\_folder** (optional, otherwise all will be in $HOME)

We suggest using **scp** on Mac/Linuxto transfer data to/from ***HPC***

> **sftp** [**your\_user\_id@hpc.udel.edu**](mailto:your_user_id@hpc.udel.edu) **(example)**

or

> **scp** [**your\_user\_id@hpc.udel.edu**](mailto:your_user_id@hpc.udel.edu)**:/home/your\_user\_id/test.txt ./**  
 input your password

1. **Clone** FUNWAVE-TVD Package

> **cd your\_funwave\_folder** (*if you created in step 1*)

> **git clone** <https://github.com/fengyanshi/Tsunami_workshop.git>

or **download** FUNWAVE-TVD Version 3.4

<https://github.com/fengyanshi/FUNWAVE-TVD/releases/tag/Version_3.4>

1. Compile the source code (DIFFERENT EXECUTABLES)

Tsunami workshop package includes four directories

/src/: source code for single-grid simulations (for most FUNWAVE applications)

/src\_mgn/: source code for multi-grid-nesting simulations

/examples/: examples used for this workshop

/doc/ : documentation for this workshop

> **cd src**

> **emacs Makefile** (if you want to modify the source file)

**NOTE:** Modify the Makefile if needed. Emacs/vi/gedit are text editors. You can use any other editor you are familiar with on the Unix/Linux system. To exit emacs editor press (Ctrl X then Ctrl C). Use FLAGs to sort out the code for your application. For example

EXEC = funwave\_cartesian

FLAG\_1 = -DDOUBLE\_PRECISION

FLAG\_3 = -DCARTESIAN

FLAG\_2 = -DPARALLEL *(if you want to run with parallel mode)*

FC = mpif90 *(on* ***amazon cloud, or full path [recommended] on your laptop****)*

*The compiled/linked executable file will be* ***funwave\_cartesian***  *inside the* ***src*** *directory*

* Compile the code for surface wave applications with flags

EXEC = funwave\_cartesian

FLAG\_3 = -DCARTESIAN

> **make clean** *(or ‘****make clobber’*** *if you want to remove the executable)*  
> **make** (or **make -f Makefile-intel** if you are building on an HPC machine with intel compiler for example)

* NOW, compile the code for spherical coordinates (e.g., basin-scale tsunami simulations)

**Change EXEC name and remove (comment out) FLAG\_3**

EXEC = funwave\_spherical

# FLAG\_3 = -DCARTESIAN

> **make clean**

> **make**

NOW, you have the executable files, funwave\_cartesian and funwave\_spherical, which will be used later in example cases.

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

**Training Session # 2**

**Topics**

* How to run FUNWAVE-TVD? Navigating the basic sections within the INPUT file for different simulations (numerics, physics, input, output, etc.).  
    
  Wiki Direct Link ⇒ <https://fengyanshi.github.io/build/html/definition.html>
* Setting up (Linux/Mac OS X and HPC machines with PBS scheduler), running, and post-processing your first FUNWAVE-TVD simulation (1D beach runup or levee overtopping with shoaling and wetting/drying).

**Practice**

1. **Solitary wave propagating on 1D Sloped Beach**



***Grid Dimensions:*** *1000X3. Grid sizes: DX=DY=1m. Depth at flat bottom: 5 m, Beach slope: 1/80.*

* Go to the directory of the 1D case

> **cd examples/Solitary\_wave\_on\_slope**

* Copy compiled executable **funwave\_cartesian** from /src/ to the current directory

> **cp ../src/funwave\_cartesian** **.**

* Check and Modify **input.txt**

The following statements are necessary in **input.txt**

**Parallel (if applicable)** PX = 2  
 PY = 1

**Depth** DEPTH\_TYPE = SLOPE  
 DEPTH\_FLAT = 5.0  
 SLP = 0.0125  
 Xslp = 500.0

**Dimensions** Mglob = 1000  
 Nglob = 3

**Time** TOTAL\_TIME = 200.0  
 PLOT\_INTV = 1.0  
 SCREEN\_INTV = 1.0

**Grid sizes** DX = 1.0  
 DY = 1.0

**Add wavemaker** WAVEMAKER = INI\_SOL  
 DEP = 5.0  
 AMP = 0.5  
 XWAVEMAKER = 250.0

**Breaking scheme (default: SWE breaker)** VISCOSITY\_BREAKING = T  
 Cbrk1 = 0.45  
 Cbrk2 = 0.35

**Wetting and Drying**MinDepth=0.01  
 **Output**

RESULT\_FOLDER = output/  
 ETA = T  
 MASK = T

* **Run the Model**

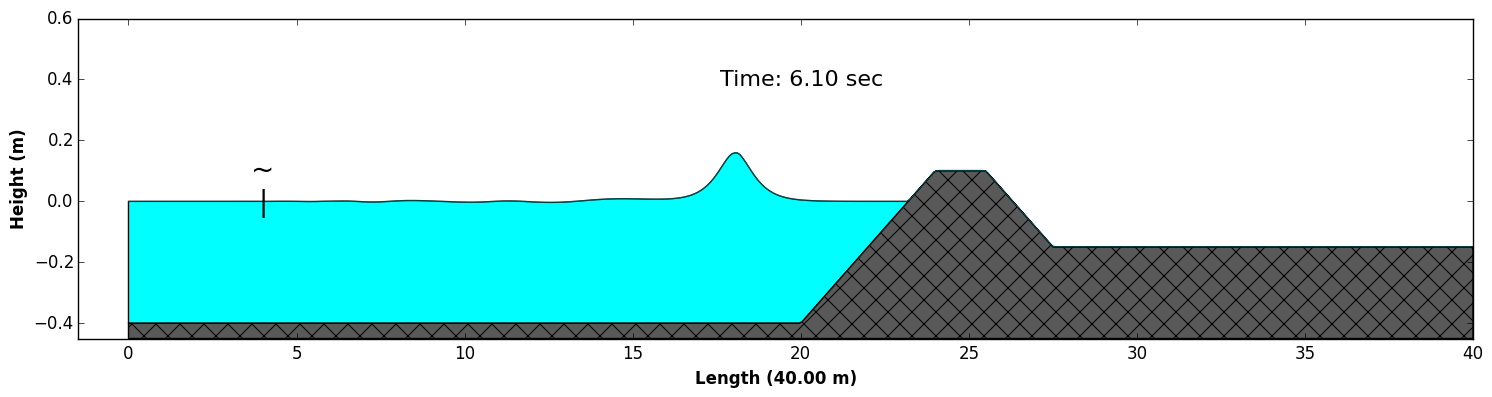
**Example of using Linux/MAC**

In the current directory

> **mpirun -np 2 ./funwave\_cartesian**

* **Post-Process Your Results**   
    
  The result is in /output/. If you use an HPC, download (or locate) your results to (on) your laptop **(eta\_#####, mask\_#####, dep.out)** and use the provided Matlab or Python scripts.

Example for using matlab:  
> **plot\_solitary**

1. **EXTRA CHALLENGE - PROGRESSION**  
     
   For those participants who were able to complete the above task and want an additional challenge, try to set up the following (in simple\_cases/levee\_1d)
   1. Modify the **input.txt** to force the wavemakers with a **Solitary Wave** as input (**Hint:** WAVEMAKER=INIT\_SOL), of **0.16 meter** amplitude.
   2. Use the supplied bathymetry (**depth\_levee.txt**) as the **‘DATA’** depth type with dimensions of [500 x 3] points.
   3. The depth at the wavemaker is **0.4** meters and it is located **4.0** meters from the left boundary.
   4. Set the **TOTAL\_TIME** to **30.0** seconds with a **PLOT\_INTERVAL** of **0.1** seconds.
   5. Activate the sponges (**DIRECT\_SPONGE** only) on the **west** (2.0m) and **east** (1.0m) side.
   6. Set the spatial discretization to **dx=dy=0.08** meters, wetting/drying to 1 millimeter (**MinDepth=0.001**), **FroudeCap=2.0**, and **CFL** condition to **0.1**.
   7. Output the **depth**, surface elevation (**eta**), and **mask**.  
        
      

**Training Session # 3**

**Topics**

* **Tsunami sources**

Use the Okada formulation to generate the initial surface deformation induced by an earthquake.

1. Input parameters
2. Source generation
3. Tsunami wave simulation

**Practice**

1. Parameters

Diagram, engineering drawing

Description automatically generated

The example source parameters can be found in /examples/okada\_source/aleutians\_source.txt

Lon Lat Len Wid Dip Rake Strike Slip(m) depth(km)

171.014400 53.305400 100 50 35.31 90.00 303.16 10.0 17.94

1. Source generation

You need a depth grid file, depth.asc, in the current directory

If you use matlab, run okada\_demo.m to generate the surface deformation saved in aleutians.txt, which has the same dimensions as the depth grid. The result of surface deformation is shown in the figure below.

Chart

Description automatically generated

1. Simulate tsunami waves

Go to /examples/Okada\_source/model/, modify input.txt if needed

The following statements are necessary in input.txt  
  
**Parallel (if applicable) … running here on 4 CPU’s (or cores/ranks)** PX = 2  
 PY = 2  
**Depth** DEPTH\_TYPE = DATA  
 DEPTH\_FILE = ../okada\_source/model\_depth.txt  
**Output folder** RESULT\_FOLDER = output/  
**Dimensions** Mglob = 600  
 Nglob = 450  
**Time** TOTAL\_TIME = 1800.0  
 PLOT\_INTV = 120.0  
 PLOT\_INTV\_STATION = 1.0  
 SCREEN\_INTV = 60.0  
**Grid sizes** Lon\_West = 165.0

Lat\_South = 45.0

Dphi = 0.03333

Dtheta = 0.03333

**Initial deformation** INI\_UVZ = T

ETA\_FILE = ../okada\_source/aleutians.txt  
**Output** ETA = T  
 Hmax = T  
  
Use the executable file funwave\_spherical.

Use the same procedures as in the previous sessions to run the model and post-process results

**Training Session # 4**

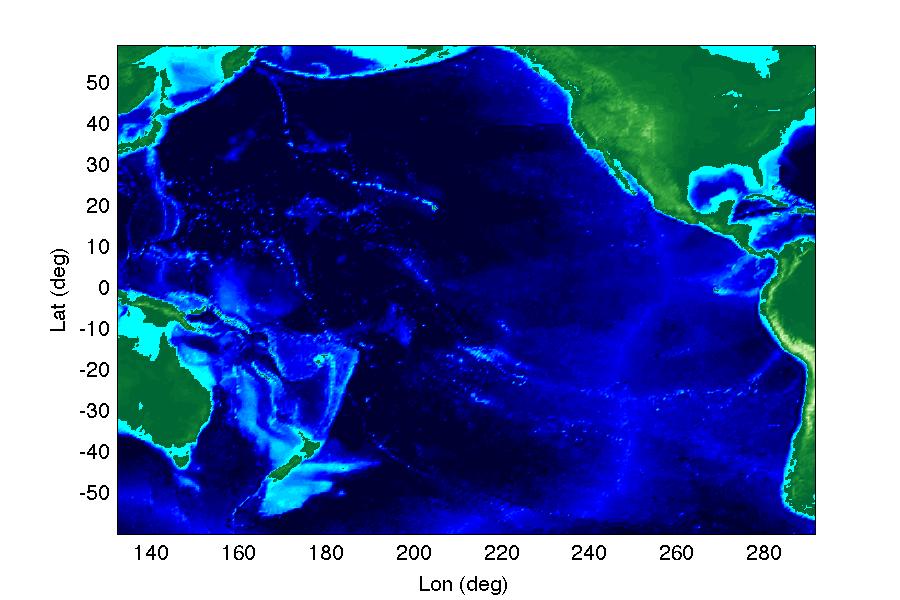
**Topic:** Tsunami simulation in a single basin-scale domain

1. Grid construction
2. Tohoku tsunami simulation (low resolution)

**Practice**

1. Grid construction using NOAA ETOPO1 Global Relief Model

<https://www.ngdc.noaa.gov/mgg/global/>



***Model configuration:*** *Computational domain covers a region of the Pacific Ocean from 60°S to 60°N in the south-north direction, and from 132°E to 68°W in the west-east direction. The example is a 30min x 30min resolution case. Grid dimensions: 320x240, Grid sizes Dphi=Dtheta=0.5 deg.*

1. Simulation

* Go to /examples/Tohoku\_tsunami\_single\_grid/work/
* copy the executable file /src/funwave\_spherical into this directory
* Check and modify **input.txt** if needed
* Run the model

The following statements are necessary in **input.txt**

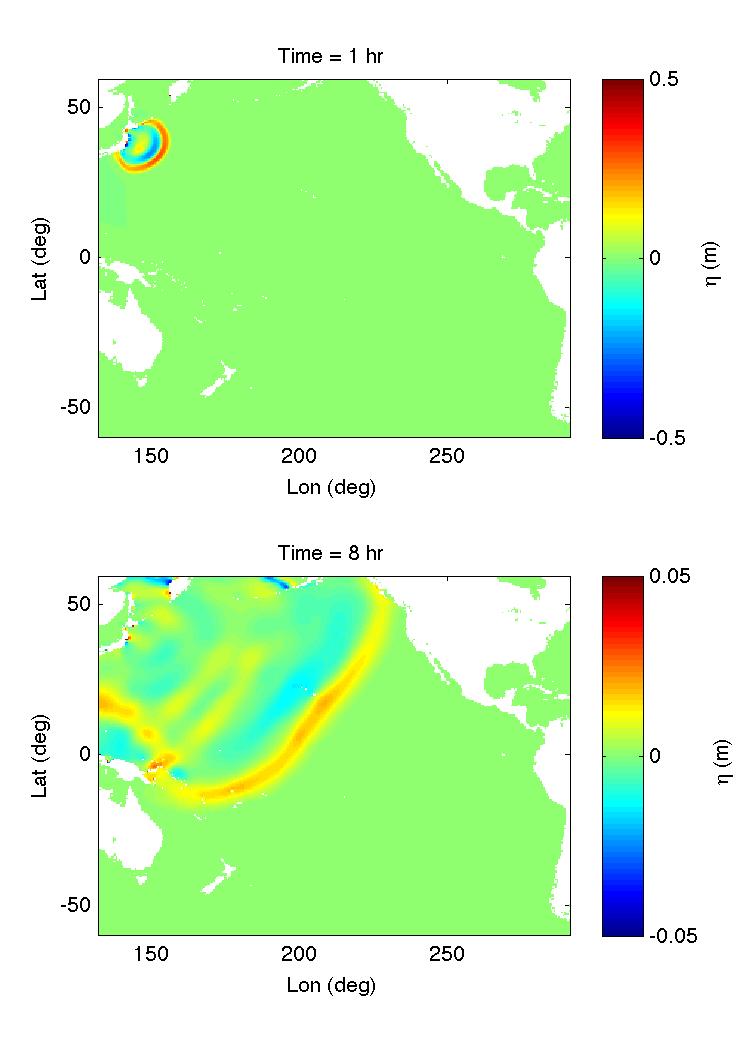
**Parallel (if applicable)** PX = 2  
 PY = 2

**Specify bathymetry** DEPTH\_TYPE = DATA  
 DEPTH\_FILE = ../external\_files/depth\_30min.txt

**Dimensions** Mglob = 320  
 Nglob = 240  
**Grid** Lon\_West = 132.0  
 Lat\_South = -60.0  
 Dphi = 0.5  
 Dtheta = 0.5  
**Time** TOTAL\_TIME = 86400.0  
 PLOT\_INTV = 3600.0  
 PLOT\_INTV\_STATION = 1.0  
 SCREEN\_INTV = 3600.0  
**Add initial conditions** INI\_UVZ = T  
 ETA\_FILE = ../external\_files/ETA\_30min.txt  
 U\_FILE = ../external\_files/U\_30min.txt  
 V\_FILE = ../external\_files/V\_30min.txt  
**Add Sponge layers** DIRECT\_SPONGE = T  
 FRICTION\_SPONGE = T  
 Sponge\_west\_width = 100000.0  
 Sponge\_east\_width = 100000.0  
 Sponge\_south\_width = 100000.0  
 Sponge\_north\_width = 100000.0  
**Add friction** Cd = 0.001  
**Avoid inundation in the basin scale (specify a large minimum depth)** MinDepth= 10.0  
**Stations/Wave Gauges** NumberStations = 78  
 STATIONS\_FILE = stations-pacific.txt  
**Output** RESULT\_FOLDER = output/   
 ETA = T  
 Hmax = T

1. Post-processing

The script can be found in /examples/Tohoku\_tsunami\_single\_grid/postprocessing/



***Figure:*** *an example of plot using Matlab code* ***plot\_surface.m***

**Training Session # 5**

**Topic:** Tsunami simulation using multi-grid-nesting

1. Multi-grid-nesting interface
2. Subgrid setup

**Practice**

1. **Compile FUNWAVE-MGN**

* Go to /src\_mgn
* Modify Makefile
* > make

1. **input.txt and subgrid\_info.txt**

****

*Figure: Basin and Hawaii sub-domain.*

**Go to /examples/NESTING\_tohoku\_2\_level/work/. In input.txt, add**

SubGrid = 1

SubGrid\_FILE = subgrid\_info.txt

TwoWayNesting = F

Check subgrid\_info.txt:

% GridDimX GridDimY Ratio\_Spacing MboxRef NboxRef

113 71 6 133 162

%Depth Subgrid (DATA or NONE)

DATA

dep\_sub\_m130\_n157\_121x79\_5min.txt

%Structure Subgrid (DATA or NONE)

NONE

1. **Tohoku tsunami in Hawaii sub-domain**

* Copy funwave\_mgn into the current directory (/work/)
* Run > mpirun -np 4 ./funwave\_mgn

1. **Postprocessing**

* Goto /postprocessing/
* Use the matlab script to generate figures

Map

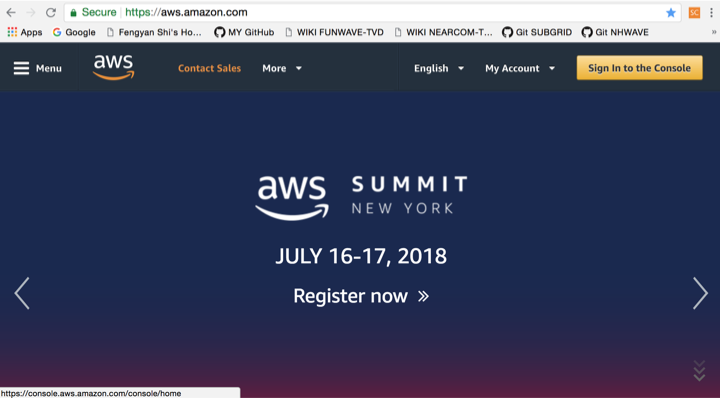
Description automatically generated

*Figure: a snapshot of surface elevation in the basin (left) and Hawaii (right) domains.*

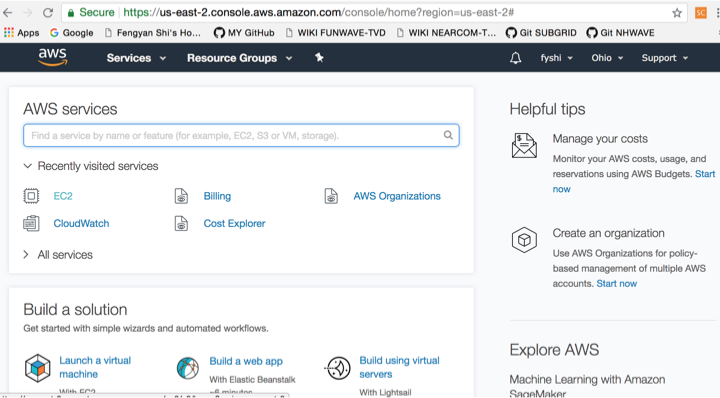
**APPENDIX:** *Amazon AWS Cloud Computing (EC2)*

Amazon AWS provides 12-month free tier <https://aws.amazon.com/free/>

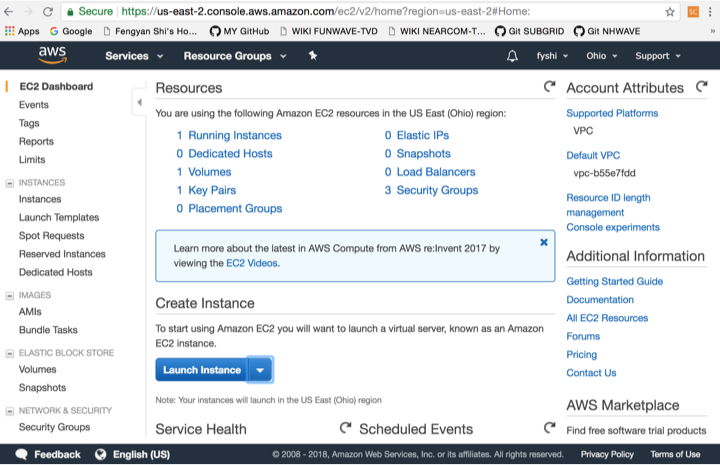
1. Sign in AWS or ‘Register now’ if you don’t have an account



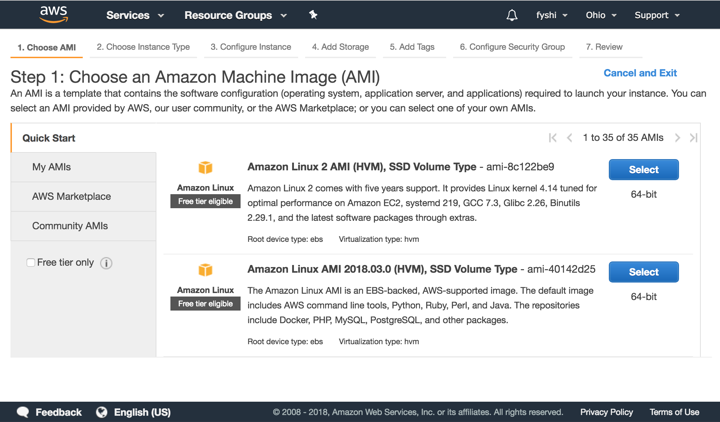
2. Click EC2 to choose EC2 service



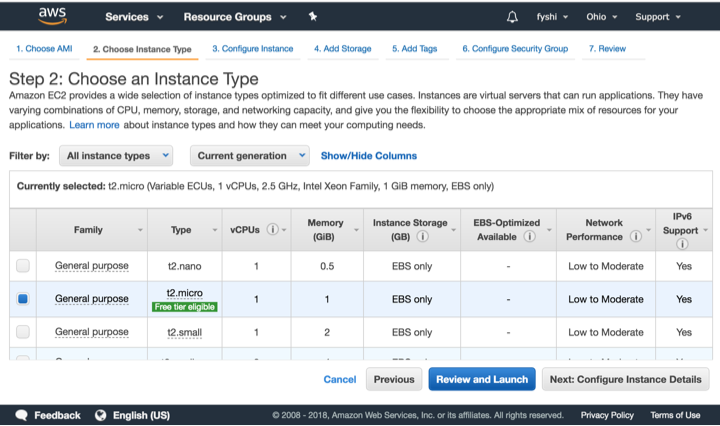
3. Create instance (build a machine for you)



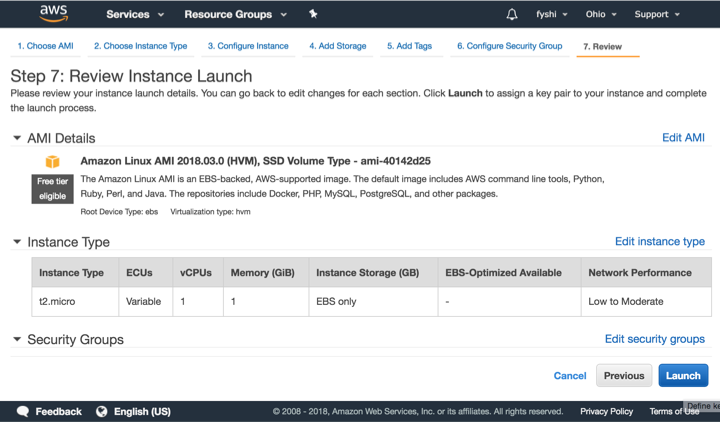
4. Select a machine (I usually choose Linux AMI 2018)



5. Select instance type (any free one)

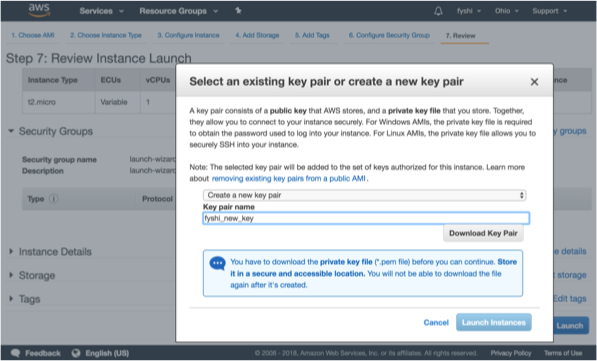


6. Click Launch

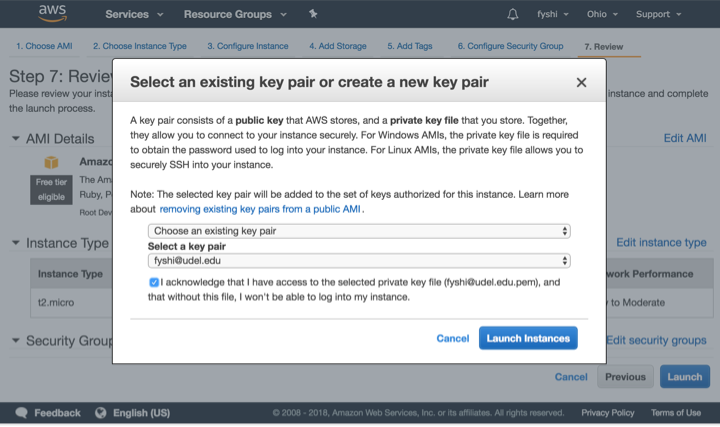


7. If you don’t have a key pair, create a new one

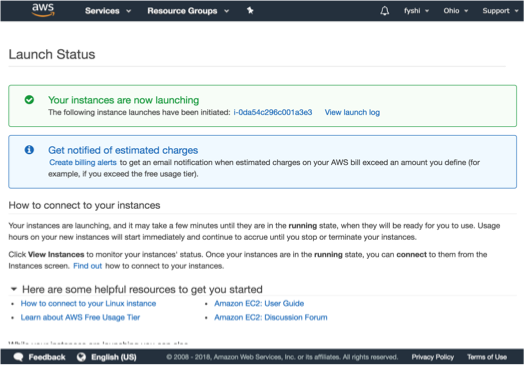
* Choose **‘create a new key pair’**
* Name it in the second line **(here I type fyshi\_new\_key)**
* Download **Key Pair**
* Store the downloaded file **(fyshi\_new\_key.pem)** into a directory you can find later
* Go to the directory**,** in command line**, type: chmod 400 fyshi\_new\_key.pem**
* You can re-use the **Key Pair** in AWS **(next page)**

****

7-A. You can re-use the Key Pair if you already had one

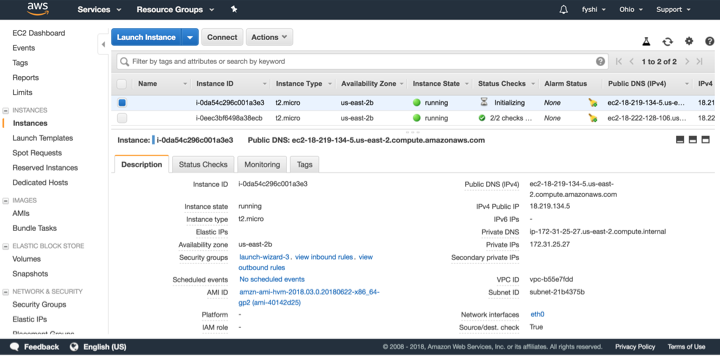
****

8. After successful launch, you will see the following page

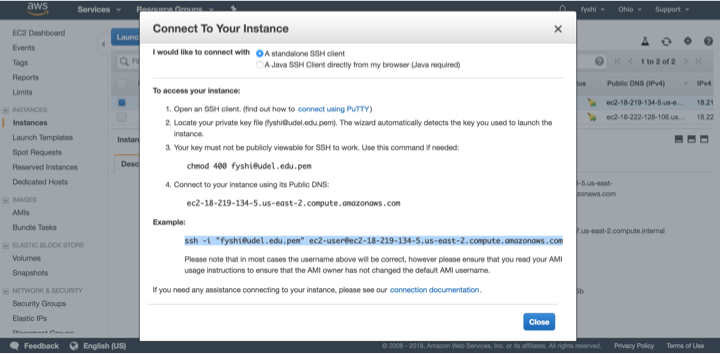


9. Click Instances on left panel, you will see instances (here I launched two) running

Choose one (blue square) you want to access by a terminal, Click Connect



10. Copy the line highlighted (paste in the terminal later)



11. Access to the virtual machine using ssh

1. Go to the directory you stored your Key pair file fyshiudeledu.pem:

$ cd directory\_you\_store\_key\_pair

1. ssh (or sftp), paste the ssh link you did in step 10:

$ ssh -i "fyshiudeledu.pem" ec2-user@ec2-18-219-134-5.us-east-2.compute.amazonaws.com

The authenticity of host 'ec2-18-219-134-5.us-east-2.compute.amazonaws.com (18.219.134.5)' can't be established.

RSA key fingerprint is 72:49:9d:11:cd:4c:70:79:8c:06:3e:6c:66:aa:59:6c.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'ec2-18-219-134-5.us-east-2.compute.amazonaws.com,18.219.134.5' (RSA) to the list of known hosts.

\_\_| \_\_|\_ )

\_| ( / Amazon Linux AMI

\_\_\_|\\_\_\_|\_\_\_|

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/

2 package(s) needed for security, out of 2 available

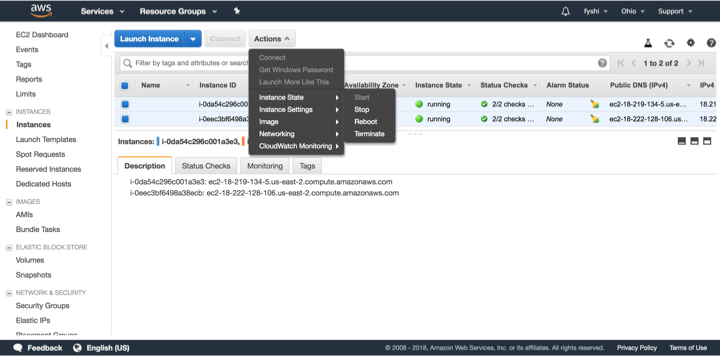
Run "sudo yum update" to apply all updates.

12. Start to use the virtual machine

13. After using it, you should stop your instances.

The “bill” will be based on the time used by instances after the free period

* To stop or terminate your instances, click Actions ->Instance state-> stop or terminate
* If you terminate your instance, everything associated with this instance will be removed, including software you installed.



13. Install **Git, gfortran, MPICH,** and **FUNWAVE-TVD**

Using the terminal that you created in step 11, go and

1. Install Git by

$ sudo yum install git

1. Install gfortran

$ sudo yum install gcc-gfortran

1. Install MPICH by

* download MPICH at <http://www.mpich.org/downloads/>
* use sftp to put the downloaded file mpich-3.2.1.tar.gz into the virtual machine
* $ tar -xzf mpich-3.2.1.tar.gz
* $ cd mpich-3.2.1/
* $ ./configure --disable-cxx
* $ make
* $ sudo make install

1. Get FUNWAVE-TVD

$ git clone <https://github.com/fengyanshi/FUNWAVE-TVD.git>

1. Compile the code and test it! And ….. Voila!