ULTRAWAVE

Installation

Python Virtual Environment

You can use the toolbox in a Python virtual environment using virtualenv:

To install virtualenv

```
pip install --upgrade pip
pip install virtualenv
```

To create a dedicated virtual env in the `ultrawave` directory (e.g., `ultrawave_venv`):

```
python -m venv ultrawave_venv
```

To activate the virtual environment

```
source ultrawave_venv/bin/activate
```

To exit the vitual env

```
deactivate
```

Conda Environment

To use the toolbox in a conda environment, you need to install either Anaconda or Miniconda.

To create a new conda environment with the name ultrawave

```
conda create -n ultrawave python=3.10
```

And activate this environment

```
conda activate ultrawave
```

To exit the environment

```
conda deactivate
```

Install UltraWave Toolbox

To build and install the `ultrawave` package in-place, such that you can edit the files without re-installing the package, run the following command at the `ultrawave` directory:

```
pip install --upgrade pip
pip install . -r requirements.txt
```

For Windows users who use conda environment, if an error shows that ‘pip’ is not recognized, you can try:

```
conda install pip
```

Run Examples

To run the jupyter notebook example, you can go to the `ultrawave` directory.

```
jupyter notebook
```

GPU Acceleration

Nvidia HPC SDK Installation

To use GPU for simulation acceleration, you need to install Nvidia HPC SDK based on your CUDA version.

After the installation, you need to set up the environment such as running this command. Reference can be taken in section 1.3 of this link.

```
export PATH=/opt/nvidia/hpc_sdk/Linux_x86_64/23.7/compilers/bin/:$PATH
```

Using One GPU

There are two ways to use GPU to accelerate your simulation.

The first way is to set environment variables inside your python script or notebook.

```
from devito import configuration
configuration['platform'] = 'nvidiaX'
configuration['compiler'] = 'pgcc'
configuration['language'] = 'openacc'
```

The second way is to set environment variables from the shell.

```
export DEVITO_PLATFORM=nvidiaX
export DEVITO_ARCH=pgcc
export DEVITO_LANGUAGE=openacc
```

Using Multiple GPUs

To use multiple GPUs, you need to set the environment variables like the one-GPU case. Besides that, the OpenMPI environment needs to be set up.

```
export PATH=/opt/nvidia/hpc_sdk/Linux_x86_64/23.7/comm_libs/mpi/bin/:$PATH
export LD_LIBRARY_PATH=/opt/nvidia/hpc_sdk/Linux_x86_64/23.7/comm_libs/mpi/lib/:$LD_LIBRARY_
```

Install `ipyparallel` and `mpi4py`:

```
pip install --upgrade pip
pip install ipyparallel
pip install mpi4py
```

If you met an error similar to this one when installing `mpi4py`, you can run

```
export CC=$(which nvc)
export CXX=$(which nvc++)
python3 -m pip install --upgrade pip setuptools wheel
CFLAGS=-noswitcherror pip install mpi4py
```

After environment setting up, you can run the python script with MPI in shell.

```
DEVITO_NPI=1 mpirun -n 4 elastic_3d.py
```