Module HAT918T
Outils de modélisation hydro-morphodynamique littorale et portuaire

Exercices de prise en main du cluster MESO@LR et applications sur SWAN 1D

Session 03

Frédéric Bouchette

Ce TD est une mise en route sur le module "outils de modélisation hydro-morphodynamique littorale et portuaire". Il suit deux objectifs.

Le premier objectif est **la prise en main de l'utilisation du cluster de calcul MESO@LR**, très représentatif des moyens de calculs lourds qu'on peut désormais trouver dans un bureau d'étude, une entreprise ou à l'université. Ce cluster est d'ailleurs accessible au niveau régional, à la fois par les universitaires et par les entreprises souhaitant mener des travaux de simulation déportée ou massive (HPC = High Performance Computing). Les approches de type HPC sont actuellement devenues incontournables dès lors qu'on s'intéresse à la fouille de données ou la mise en place d'algorithmes d'intelligence artificielle (IA). Dans ces stratégies, la quantité de donnée produite (ici numérique) est primordiale : elle garantit a) la qualité de l'apprentissage pour l'IA, b) la qualité des lois établies en probabilité ou c) la robustesse de toute autre méthode permettant d'obtenir – à partir d'un grand ensemble de calculs réalisés sous certains forçages – de nouveaux résultats pour des conditions quelconques sans avoir à les calculer. En pratique, le premier objectif de ce TD est de donner quelques bons réflexes d'utilisation d'un cluster de calcul afin d'ouvrir votre esprit aux techniques de fouille de données.

Le second objectif est **la prise en main d'un premier outil de simulation numérique** classique en hydro-morphodynamisme littoral : SWAN (pour Simulating WAves Nearshore) [1] qui est un produit libre et open-source de l'université de Delft [2] aux Pays-Bas. Ce modèle est une des références pour la modélisation numérique de la propagation de l'action des vagues du large vers la côte. L'action des vagues est une grandeur physique intimement liée à la densité spectrale d'énergie des vagues, elle-même directement contrôlée par la superposition linéaire des oscillations de la surface de la mer à toutes les fréquences. SWAN permet de calculer des spectres de vagues, et des grandeurs moyennes caractéristiques comme la hauteur significative, la période pic, etc... C'est un modèle 2DH, c'est à dire qu'il produit en sortie des variables qui existent sur un domaine horizontal (l'emprise cartographique du modèle). L'objectif de ce TD n'est pas de poser le formalisme physique derrière le fonctionnement de SWAN, mais de commencer par mettre les mains dans le cambouis pour acquérir une compétence de base en simulation numérique des vagues en zone littorale. En pratique, le TD met en oeuvre SWAN dans un mode 1D, c'est à dire le long d'un simple profil du large vers la côte, et se concentre sur l'analyse de l'atténuation des vagues sur différents profils et types de fonds marins. Il propose à la fin de définir des lois permettant d'extrapoler les résultats des calculs obtenus.

Bons premiers pas dans le merveilleux monde du calcul numérique !

1. `https://swanmodel.sourceforge.io`
2. `https://www.tudelft.nl/en/ceg/about-faculty/departments/hydraulic-engineering/sections/environmental-fluid-mechanics/research/swan/`

1 Utilisation du cluster de calcul MESO@LR

1.1 Avant de partir...

Pour utiliser le cluster de calcul MESO@LR :

- il vous faut un identifiant pour se connecter sur le cluster MESO@LR. Dans la suite, on appellera cet identifiant `e_GCL-XX` et vous devrez remplacer ce terme par celui qui vous a été communiqué. Vous devez disposer également d'un mot de passe associé à cet identifiant ;

- il vous faut un **terminal** sur votre ordinateur personnel ou celui que vous utilisez pour travailler. Un terminal (ou shell) est un logiciel simple d'interaction avec le système qui vous permet de taper en ligne de commande des ordres et vous connecter en SSH (Secure SHell) sur un autre ordinateur. Si vous êtes sous Linux ou Mac [3], vous disposez d'un tel outil nativement. Si vous êtes sous Windows 10, vous devez installer un tel outil [4]. Dans la suite, à chaque fois qu'une commande devra être tapée dans le terminal/shell, les lignes concernées seront arbitrairement précédées du terme `\$` qui représentera l'invite de commande (à ne pas retaper), c'est à dire le signal affiché par le terminal disant que vous pouvez envoyer des ordres au système. La forme que prend cette ligne de commande dépend de la configuration de votre environnement personnel. Cela peut être par exemple `[GCL_XX@muse-login01]$` ;

- Pour utiliser confortablement un terminal en ligne de commande – sur votre propre ordinateur ou sur le cluster – il est primordial que vous maîtrisiez un minimum un langage de shell comme `Bash` [5]. Entrainez vous régulièrement et apprenez par coeur le fonctionnement d'une bonne vingtaine de commandes, y compris celles-ci (en vert des mot-clés réservés, en noir des commandes externes) : `man`, `more`, `ls`, `pwd`, `cat`, `rmdir`, `mkdir`, `cd`, `touch`, `chmod`, `tree`, `touch`, `nano`, `vi`, `touch`, `which`, `grep`, `find`, `alias`, `echo`, `read`, `if`, `for`, `case`, `while`, `exit`, `git` et les expressions régulières et le traitement des chaînes de caractères. Il vous sera également très utile d'apprendre à créer vos propres petits scripts en `Bash` afin d'automatiser vos travaux [6] ;

- vous devez disposer d'une connexion internet, qui vous permet à partir de votre ordinateur personnel ou de travail, de vous connecter sur le cluster et de préparer et commander à distance les calculs que vous souhaitez lancer. Toutes les opérations se déroulent avec une interface textuelle en ligne de commande. Vous devez renoncez à afficher vos résultats de calcul directement à partir du cluster (c'est possible, mais nous n'utiliserons pas la technologie). Vous devez donc être en mesure d'envoyer et récupérer des fichiers sur le cluster confortablement à partir de votre ordinateur de travail. Les entrées/sorties sur le cluster sont souvent un point délicat à gérer et une source de problèmes. Donc posez vous toujours la question suivante : que doit-on traiter sur le cluster et que doit-on traiter en local ? Le corollaire de cette question étant : qu'est-on obligé de faire transiter par internet entre notre ordinateur local et le cluster ?

3. `https://www.macplanete.com/tutoriels/26348/client-ssh-mac-connexion-serveur-machine`
4. `https://www.commentcoder.com/terminal-windows/`
5. `https://fr.wikibooks.org/wiki/Programmation_Bash/Notions_essentielles_du_shell_bash`
6. `https://ftp.traduc.org/doc-vf/gazette-linux/html/2006/133/lg133-A.html` ou `https://doc.ubuntu-fr.org/tutoriel/script_shell`

- Enfin, vous devez disposer d'un environnement de programmation sous `Python` sur votre ordinateur local, afin de pouvoir préparer des fichiers à envoyer au cluster, et traiter les résultats des sorties de simulations issues du cluster. Sur le cluster lui-même, vous pourrez également lancer différentes versions de `Python`. Vous vous reporterez aux documents transmis dans mes cours pour l'installation de `Python` via `Conda` et la configuration optimale de ce genre d'environnement. Sur le cluster, l'installation est déjà faite.

1.2 Première connexion sur le cluster

Pour vous connecter sur le cluster, vous devez ouvrir un terminal avec la possibilité d'utiliser SSH, et vous connecter avec votre identifiant et votre mot de passe sur une des deux machines d'accès au cluster qui sont `muse-login01.hpc-lr.univ-montp2.fr` et `muse-login02.hpc-lr.univ-montp2.fr`. Par exemple :

```
1   \$ ssh e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr
```

Vous utiliserez indifféremment `muse-login01` ou `muse-login02` selon que vous constaterez que l'une ou l'autre des machines est lente ou rapide (cela dépend du nombre de personnes connectées, et de leur bon comportement, on y reviendra). Une fois votre mot de passe tapé, vous êtes dans un terminal/shell sur le cluster, c'est à dire que vous disposez d'une fenêtre de commande ouverte sur un ordinateur déporté. Vous pouvez interagir avec le système à distance :

```
1   \$ ls
2   install-softs  intel  privatemodules  ...  work_f_gcl
3   \$ cd work_f_gcl
4   \$ ls
5   \$ touch toto
6   \$ ls
7   toto
8   \$ rm toto
9   \$ ls
10  \$
```

Vous disposez d'un répertoire personnel `~` sur lequel vous pouvez toujours revenir en tapant `cd` ou `cd /home/e_gcl-XX`. Dans ce répertoire, vous avez plusieurs sous-répertoires importants pour la suite (ils sont listés avec la commande `ls`). Vous constaterez que vous avez tous en commun le répertoire `work_f_gcl` qui est partagé à l'échelle du groupe `gcl`, c'est à dire tous les comptes en `e_gcl-XX`. Si vous regardez dans ce répertoire, vous constatez que vous pouvez y partager du matériel numérique :

```
1   \$ cd ~/work_f_gcl
2   \$ touch gcl-XX-monfichiertest.txt
3   \$ ls
```

Essayez d'utiliser les commande `chmod` sur vos différents fichiers et observez ce qui se passe. Après quelques minutes d'échanges entre vous, pensez à supprimer le fichier créé :

```
\$ cd ~/work_f_gcl
\$ rm gcl-XX-monfichiertest.txt
```

Une règle fondamentale de l'utilisation du cluster est que vous devez maintenir le répertoire `work_f_gcl` aussi propre que possible. Pensez que vous avez potentiellement des droits sur les fichiers que vous créez que les autres n'ont pas ; et donc vous pouvez rapidement transformer le répertoire en une vaste décharge. Une organisation en répertoires et sous-repertoires parfaite est de rigueur.

Vous constaterez que vous avez aussi dans votre répertoire personnel un sous-répertoire `scratch` qui a un statut très particulier : c'est répertoire que vous voyez depuis les machines `muse-login01` et muse-login01, mais qui est en fait directement rattaché aux noeuds de calculs du cluster. A ce titre, tout ce qui va être stocké sur ce répertoire `scratch` va pouvoir être écrit ou lu par les noeuds de calcul de manière très rapide. En pratique, vous allez devoir stocker dans ce répertoire `scratch` tout ce qui a un coût en temps machine et qui est directement associé aux processus de lecture-écriture pendant les simulations. Notez que le répertoire `scratch` de votre environnement personnel pointe en fait sur un sous-répertoire du répertoire général `lustre` comme vous pouvez le constater en faisant par exemple :

```
\$ cd
\$ ls -l
...
lrwxrwxrwx  1 root     root    16 14 avril 16:45 scratch -> /lustre/e_gcl-XX
...
```

Autrement dit, vous ne partagez pas du tout le contenu de votre répertoire `scratch` avec les autres utilisateurs, même au sein de votre groupe `gcl`.

Enfin, vous remarquerez qu'aussi bien `scratch` que `work_f_gcl` sont plus exactement des liens symboliques, c'est à dire des noms utilisés à un endroit de l'arborescence pour pointer sur un autre lieu de l'arborescence. De manière générale, vous pouvez utiliser les liens symboliques avec la commande `ln -s`.

1.3 Automatiser la connexion sur le cluster avec une clé SSH

Vous allez rapidement en avoir assez de taper un mot de passe complexe pour vous connecter sur le cluster, d'autant que la saisie du mot de passe va également être nécessaire lorsque vous voudrez envoyer des fichiers ou les récupérer avec les commandes que nous verrons plus loin. Une manière de simplifier la connexion (sous entendu l'obtention du droit d'envoyer et recevoir des contenus, que ce soit des ordres ou des informations/données) est de s'appuyer sur un système de type clé `SSH` (Secure SHell). Le principe du SSH est assez complexe[7] et sort du cadre de ce TD, mais le principe est de sécuriser un échange en utilisant une clé publique et une clé privée qui sont liées par un algorithme de criptage très difficile à violer. Une implémentation de cette technologie est `OpenSSH`, qui dispose de différents outils pour réaliser des connexions et échanges d'information dans le cadre du protocole `SSH`. La manière de mettre en oeuvre `SSH` dans votre environnement dépend si vous êtes sous Linux, Windows ou MAC. Vous vous reporterez aux sites suivants selon votre installation :

7. `https://fr.wikipedia.org/wiki/Secure_Shell`

Linux : `https://blog.microlinux.fr/cle-ssh/`

Windows : `https://docs.microsoft.com/fr-fr/windows-server/administration/openssh/openssh_keymanagement`

Mac : `https://mesocentre.univ-amu.fr/ssh/` (marche aussi pour les autres)

De manière générale, vous pourrez utiliser une `passphrase` afin d'augmenter la sécurité. Le problème est que vous devrez la taper à la place du mot de passe, ce qui va ralentir un peu vos opérations. C'est donc un choix à faire : soit vous la donnez et vous devrez la taper, soit vous la mettez à `Null` (tapez directement entrée lorsqu'elle vous est demandée) et vous pourrez vous connecter directement. Dans la suite, on part du principe qu'aucune `passphrase` n'a été saisie.

Si on prend l'exemple d'un système linux, toute l'opération peut se faire dans le terminal de votre ordinateur local :

```
  \$ ssh-keygen -t rsa -b 16384
  ...  # ne pas saisir de pass phrase
  \$ ssh-copy-id -i .ssh/id_rsa.pub e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr
  /usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/id_rsa.pub"
  /usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to ...
  /usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are ...
  # Saissisez votre mot de passe, le système affichera:
  Number of key(s) added: 1
  Now try logging into the machine, with:   "ssh 'e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr'"
  and check to make sure that only the key(s) you wanted were added.
  \$ ssh e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr
```

Vous constaterez que vous vous connectez sans avoir à produire le mot de passe, ce qui ne veut pas dire que votre connexion n'est pas sécurisée. Elle l'est depuis l'ordinateur local.

1.4 Déposer et rapatrier du matériel sur le cluster

Comme on l'a vu plus haut, le cluster se commande à distance, via internet. Mais donner des ordres ne suffit pas. Il faut pouvoir déposer sur l'arborescence de fichiers du cluster des logiciels, des fichiers de configuration/forçages, des fichiers de données, et récupérer les résultats des calculs, entre autres choses. Pour cela, on dispose d'outils utilisant le même protocole `OpenSSH` que pour la connexion.

La commande la plus importante est `SCP` (Secure CoPy). Elle est disponible par défaut sur le cluster, et si vous avez installé convenablement `OpenSSH` sur votre ordinateur local, elle devrait être également disponible. Pour le vérifier, faites depuis votre terminal local :

```
  \$ cd
  \$ cat > myfile << EOF
ceci est un fichier qui va partir sur le cluster
EOF
  \$ scp myfile e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr:~
  myfile                                100%   15     0.5KB/s   00:00
```

où `myfile` est un fichier qui a été créé par la commande `cat`. Alternativement, on aurait pu utiliser `touch` ou un éditeur comme `nano` ou `vi`. Toutes ces commandes sont disponibles

sur le cluster comme sur votre environnement local (sauf quelquefois sous windows). Avec la commande `scp` utilisée ci-dessus, votre fichier `myfile` est envoyé dans votre répertoire personnel `~` sur le cluster. Pour le vérifier, vous pouvez vous connecter avec `ssh` sur le cluster et taper dans votre terminal distant :

```
  \$ cd
  \$ more myfile
ceci est un fichier qui va partir sur le cluster
  \$ nano myfile   # vous éditez alors le fichier sur le cluster
```

Votre fichier a bien été copié sur le cluster et il a été modifié sur place.

Vous pouvez utiliser `scp` pour copier des répertoires entiers. Il suffit pour cela d'utiliser l'option `-r` (attention il y a des pièges, à voir à l'usage) :

```
  \$ cd
  \$ mkdir TEMP
  \$ scp -r TEMP e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr:~
```

Dans le cas du fichier `myfile` comme du répertoire `TEMP` ci-dessus, le matériel est envoyé sur le cluster dans le répertoire personnel `~`. Si vous indiquez après les `:` une autre destination, les fichiers ou répertoires seront copiés à cet endroit là. Vous devrez faire attention à l'existence du répertoire distant et au fait que vous copiez un répertoire local et son contenu ou uniquement son contenu.

Vous pouvez bien sur copier des fichiers (et des répertoires et leur contenu) du cluster vers votre ordinateur local en faisant simplement, à partir de votre terminal local :

```
  \$ scp -r  e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr:~ TEMP
```

Pratiquez plusieurs minutes des copies diverses entre votre ordinateur local et votre répertoire personnel sur le cluster (dans les deux sens, en éditant les fichiers de chaque coté et en les copiant dans un sens et dans l'autre. Ne pratiquez pas dans `work_f_gcl` mais uniquement dans votre répertoire personnel.

Il y a de nombreuses autres commandes possibles utilisant SSH, pour faire exécuter à distance certaines opérations simples ou complexes. Mais avec `ssh` et `scp`, vous pouvez vous sortir de à peu près toutes les situations que vous rencontrerez en simulation numérique HPC. Nous reviendrons sur les commandes `SSH` dans un TD ultérieur.

2 Déployer `SWAN` sur le cluster

Il n'y a pas de manière unique de déployer un logiciel comme SWAN sur un ordinateur local ou sur un cluster. On peut télécharger les sources du code et tout compiler à la main. On peut utiliser des outils d'aide à la compilation qui sont développées par les développeurs du code. Mais dans un contexte d'utilisation sur un cluster comme MESO (et sur les machines personnelles également), on peut aussi avoir recours à un outil d'aide à l'installation optimisé pour le calcul scientifique, c'est à dire prenant soin de regrouper au sein du binaire compilé des

librairies dûment choisies et correctement liées entre elles. C'est le choix qui est fait dans ce TD.

2.1 Installation de PAGURE

La première étape est d'installer un logiciel appelé PAGURE[8], développé par un ancien doctorant et post-doctorant de GLADYS (Fabien Rétif[9]). Ce logiciel va récolter l'ensemble des librairies nécessaires au fonctionnement de SWAN, et les lier dans leur bonne version avec le compilateur choisi pour compiler sur le cluster.

On se connecte d'abord sur le cluster :

```
1  \$ ssh e_gcl-XX@muse-login01.hpc-lr.univ-montp2.fr
```

A partir d'ici, toutes les commandes qui sont présentées dans cette section sont à saisir sur le cluster, et non pas sur votre machine locale. On utilise le logiciel `git` – qu'on verra en détails plus tard – pour récupérer pagure.

```
1  \$ cd
2  \$ mkdir install-softs
3  \$ cd install-softs
4  \$ mkdir pagure.git
5  \$ cd pagure.git
6  \$ git clone https://github.com/fretif/pagure.git .
7  Cloning into '.'...
8  remote: Enumerating objects: 2000, done.
9  remote: Counting objects: 100% (334/334), done.
10 remote: Compressing objects: 100% (231/231), done.
11 remote: Total 2000 (delta 229), reused 200 (delta 103), pack-reused 1666
12 Receiving objects: 100% (2000/2000), 4.67 MiB | 0 bytes/s, done.
13 Resolving deltas: 100% (1475/1475), done.
14 \$ ./pagure.sh   # affiche un message de PAGURE par défaut
```

`PAGURE` est désormais installé dans le répertoire `/install-softs/pagure.git`.

Avant de lancer `PAGURE` pour réaliser le télé-chargement, l'installation des librairies et la compilation totale de SWAN, il faut installer dans l'environnement du cluster le compilateur qui va permettre ces opérations. On utilise pour cela le logiciel `module` (et oui drôle de nom !) qui est très courant dans les environnements cluster où on doit changer de configuration de compilation assez souvent. On détaillera ce logiciel plus tard. Pour l'instant, on se contente d'en utiliser quelques fonctionnalités, toujours sur le cluster, dans votre compte :

```
1  \$ module purge
2  \$ module load cv-standard use.own intel/compiler/64/2017.1.132
```

On a ainsi chargé le compilateur de marque `Intel` dans sa version 2017, disponible sur le cluster comme on peut le voir si on fait un `module avail` en ligne de commande.

Une fois ceci fait, on peut procéder au lancement de `PAGURE` :

8. `https://github.com/fretif/pagure`
9. `https://www.linkedin.com/in/fabienretif/`

```
\$ echo "check_certificate = off" >> ~/.wgetrc
\$ ./pague.sh --prefix=/home/e_gcl-XX/softs --system=cluster
   --compiler=intel --filter=SWAN --module-dir=/home/e_gcl-XX/privatemodules
# un temps looooooooooooooooooooooonnnnnnnnnng! (sans doute 1h la première fois), puis:
[  INFO  ]  Removing archive file and source files
[  OK  ]  Install swan 41.31
[  OK  ]  Congratulation, you did it
```

La première commande permet simplement de contourner des questions de certification du téléchargement de certaines librairies (`mpich`). La seconde commande va compiler et installer un ensemble de modules (au sens du logiciel `module` dans le répertoire `privatemodules` de votre environnement personnel, télécharger le logiciel `SWAN` et le compiler et lier à ces librairies, puis l'installer dans le répertoire `soft` de votre environnement personnel. La compilation peut durer un certain temps. Donc vous avez la possibilité d'aller boire un café. Il est aussi possible qu'elle plante : ce sont les joies du déploiement de logiciels complexes sur architecture HPC. Il faut accepter cela et garder en tête que le déploiement est une spécialité. Vous serez sans doute de "simples" utilisateurs de codes déjà installés dans votre carrière, même si vous avez l'ambition de faire du HPC. Toutefois, en maîtrisant `PAGURE` et quelques autres outils, vous pourrez prétendre à installer plusieurs outils de modélisation océanographique sur à peu près n'importe quel environnement.

2.2 Préparation de l'environnement pour une utilisation de SWAN

Une fois la compilation terminée, vous avez une version de `SWAN` correctement compilée et stockée dans votre environnement, ainsi que de nombreuses librairies accessible via le logiciel `module`. Mais tout cet écosystème n'est pas chargé en mémoire par défaut ; en effet, on ne charge que des librairies sur demande pour préserver la mémoire, et on configure ab minima l'environnement pour éviter toute complexité inutile. Donc, pour préparer votre environnement à l'utilisation de `SWAN` (dans sa version finale 41.32), vous devez – à chaque fois que vous vous connectez sur un nouveau terminal sur le cluster – simplement évoquer le module correspondant au chargement de `SWAN` :

```
\$ module purge
\$ module load cv-standard use.own swan
```

Vous pouvez automatiser ce chargement en plaçant ces commandes dans le fichier `.bashrc` (situé dans votre répertoire personnel). Mais je le déconseille dans la mesure où plus tard, vous utiliserez des outils numériques différents qui vont nécessiter le chargement en début de session de librairies et configurations différentes. Donc pensez à les taper avant d'utiliser SWAN.

Il faut enfin vérifier qu'on peut facilement accéder à l'exécutable du logiciel `SWAN`, qui s'appelle `swanrun`, et qui pointe sur un binaire `swan.exe`. L'exécutable est stocké dans le répertoire `/softs/swan/mpich321/icc17/41.31/bin`. Dans ce répertoire, on a deux binaires `swan_mpi.exe` et `swan_omp.exe`. Dans notre cas, nous utilisons le binaire `MPI` donc nous faisons :

```
\$ cd ~/softs/swan/mpich321/icc17/41.31/bin
\$ cp swan_mpi.exe swan.exe
```

Par ailleurs, si on fait sur la ligne de commande :

```
\$ echo \$PATH
/home/e_gcl-18/softs/swan/mpich321/icc17/41.31/bin:/home/e_gcl-18/softs
/netcdf/hdf5.110/mpich321/icc17/fortran/4.5.3/bin:/home/e_gcl-18/softs/
netcdf/hdf5.110/mpich321/icc17/c/4.8.0/bin:/home/e_gcl-18/softs/parallel
-netcdf/mpich321/icc17/1.12.1/bin:/home/e_gcl-18/softs/hdf5/mpich321/icc
17/1.10.5/bin:/home/e_gcl-18/softs/mpich/icc17/3.2.1/bin:/trinity/shared/
apps/cv-standard/intel/2017.1.132/compilers_and_libraries_2017.1.132/linux
/bin/intel64:/trinity/shared/apps/cv-standard/intel/2017.1.132/compilers_
and_libraries_2017.1.132/linux/bin:/trinity/shared/apps/cv-standard/intel
/2017.1.132/debugger_2017/gdb/intel64:/trinity/shared/apps/cv-standard/in
tel/2017.1.132/debugger_2017/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:
/usr/sbin:/opt/dell/srvadmin/bin
```

On voit que le chemin contenant `swanrun` et `swan.exe` est bien compris dans la variable `$PATH`, ce qui signifie que le modèle numérique est visible de partout dans votre environnement.

Pour tester la bonne marche de `SWAN`, vous allez simplement lancer l'exécutable :

```
\$ swanrun
**ERROR: no name SWAN input file given!
Usage: swanrun -input file [-omp n | -mpi n]
```

Le logiciel râle parce que vous n'avez pas fourni suffisamment d'information, mais il est bien là.

2.3 Calcul test avec `SWAN 1D` sur le cluster

On va terminer cette section de mise en place de `SWAN` en utilisant les données qui sont stockées dans `work_f_glc/SWAN1D/common_stuff`. Vous copiez les fichiers `maupiti1D_1m.swn` et `MaupitiBathy1D_SWAN_1m.dat` dans un répertoire que vous aurez créé dans votre arborescence, et vous lancez correctement `SWAN` :

```
\$ cd
\$ mkdir SWAN1D_temp
\$ cd SWAN1D_temp
\$ cp ~/work_f_gcl/SWAN1D/common_stuff/maupiti1D_1m.swn .
\$ cp ~/work_f_gcl/SWAN1D/common_stuff/MaupitiBathy1D\_SWAN\_1m.dat .
\$ swanrun -input maupiti1D_1m.swn
```

Le premier fichier d'extension `.swn` est le fichier de configuration. Le second fichier d'extension `.dat` est le fichier contenant le profil bathymétrique utilisé pour cette simulation test. Vous pouvez changer librement les noms des répertoires, les noms de fichiers mais attention : les noms de fichiers et les chemins vers ces fichiers sont utilisés dans le fichiers de configuration de `SWAN` ; donc il faudra reporter tout changement dans ces fichiers. Vous pouvez d'ailleurs jeter un coup d'oeil à ces 2 fichiers en les parcourant avec `more` ou un des éditeurs en ligne de commande `vi` ou `nano`.

Une fois que vous avez lancé `swanrun`, vous devez avoir obtenu ce genre de sortie :

```
swan.exe est /home/e_gcl-18/softs/swan/mpich321/icc17/41.31/bin/swan.exe

SWAN is preparing computation

 iteration    1; sweep 1
+iteration    1; sweep 2
+iteration    1; sweep 3
+iteration    1; sweep 4
not possible to compute, first iteration

 iteration    2; sweep 1
+iteration    2; sweep 2
+iteration    2; sweep 3
+iteration    2; sweep 4
accuracy OK in  24.67 % of wet grid points ( 99.50 % required)

 iteration    3; sweep 1
+iteration    3; sweep 2
+iteration    3; sweep 3
+iteration    3; sweep 4
accuracy OK in   0.07 % of wet grid points ( 99.50 % required)
[...]
```

Vous remarquerez aussi que dans le répertoire dans lequel vous avez lancé le code, vous trouvez maintenant plusieurs nouveaux fichiers. Question : est-ce que la simulation s'est bien déroulée ? Pour cela, regardez toujours le contenu du fichier `swaninit` et de ceux d'extension `.erf` et `.prt`. Commentez tout cela en classe.

Bravo. Vous avez un outil `SWAN` opérationnel.

3 Un problème d'analyse de l'atténuation des vagues avec `SWAN`

3.1 Présentation rapide de la physique de SWAN

Voir au tableau :-)

3.2 Présentation de la configuration de calcul

Vous allez récupérer une archive envoyée par l'enseignant, qui contient :

- des configurations pour lancer le modèle (fichier de bathymétrie `.dat` et fichier de configuration `.swn`)
- des scripts python qui permettent de préparer un fichier de profil bathymétrique, et des sorties graphiques à partir des fichiers résultats
- de petits fichiers contenant des commandes pour envoyer/récupérer les différents fichiers liés à l'utilisation de `SWAN` sur le cluster.

Avec ce matériel type, vous allez :

- penser un projet de simulation d'un profil de votre choix (plage sableuse, ouvrage portuaire, récif coralligène, NBS, etc ...),
- construire le fichier de configuration avec l'aide de l'enseignant,
- construire un fichier de bathymétrie (avec un script python adapté),
- déployer votre configuration sur le cluster et faire des tests de calculs,
- représenter les premiers résultats rapatriés sur votre ordo local,
- mettre en place une stratégie d'analyse du comportement de votre profil en terme de dissipation des vagues sous différentes conditions
- penser une manière d'automatiser votre effort de calcul sur le cluster.

BON COURAGE :-)