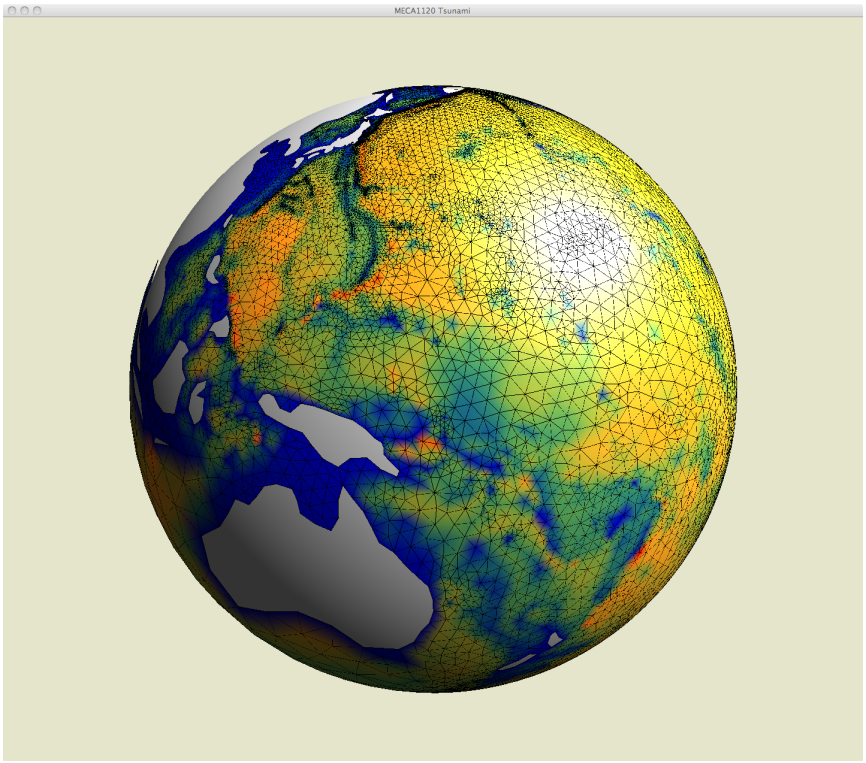
1 Modélisation d'un tsunami par éléments finis



L'objectif du projet est de vous initier aux difficultés de la mise au point et de la certification d'une application numérique. Rassurez-vous : il ne s'agit nullement de vous transformer en experts de l'architecture de grandes applications numériques ! Nous nous limiterons à l'écriture d'un tout petit programme `python` pour simuler le tsunami qui a dévasté le Japon en 2011.

1.1 La projection stéréographique...

Pour résoudre un problème sur la sphère, nous allons considérer une projection stéréographique de tous les points de l'océan sur un plan. Si le rayon de la terre est R et le centre de la terre est à l'origine, les coordonnées stéréographiques (x, y) de tout point (x_*, y_*, z_*) de la surface terrestre sont définies par les expressions suivantes. Le changement de variable inverse se déduite aisément en tenant compte que $(x_*^2 + y_*^2 + z_*^2) = R^2$.

$$x = \frac{2Rx_*}{R+z_*} \qquad x_* = \frac{4R^2x}{4R^2+x^2+y^2}$$

$$y = \frac{2Ry_*}{R+z_*} \qquad y_* = \frac{4R^2y}{4R^2+x^2+y^2}$$

$$z_* = \frac{(4R^2-x^2-y^2)R}{4R^2+x^2+y^2}$$

L'intérêt de ce changement de variables est de pouvoir résoudre les équations des eaux peu profondes sur

un plan et non sur la surface de notre bonne vieille planète. Evidemment, il faudra légèrement modifier les équations pour tenir compte de la forme sphérique...

1.2 Les équations à résoudre...

Les équations des eaux profondes linéaires s'écrivent pour les coordonnées stéréographiques :

$$\begin{cases} \dfrac{\partial \eta}{\partial t} + \left(\dfrac{4R^2+x^2+y^2}{4R^2}\right)\dfrac{\partial}{\partial x}\Big(hu\Big) + \left(\dfrac{4R^2+x^2+y^2}{4R^2}\right)\dfrac{\partial}{\partial y}\Big(hv\Big) &= \dfrac{(xu+yv)h}{2R^2} \\ \dfrac{\partial u}{\partial t} + \left(\dfrac{4R^2+x^2+y^2}{4R^2}\right)\dfrac{\partial}{\partial x}\Big(g\eta\Big) &= -\gamma u + fv \\ \dfrac{\partial v}{\partial t} + \left(\dfrac{4R^2+x^2+y^2}{4R^2}\right)\dfrac{\partial}{\partial y}\Big(g\eta\Big) &= -\gamma v - fu \end{cases}$$

Les inconnues η et $\mathbf{u} = (u, v)$ représentent respectivement l'élévation du niveau de la mer par rapport à une hauteur de référence ($z = 0$) et la vitesse horizontale moyenne sur la colonne d'eau. Les deux dernières équations expriment la conservation de la quantité de mouvement, tandis que la première relation provient de la conservation de la masse d'eau. Pour terminer la description du modèle, il reste à définir un certain nombre de paramètres matériels ou de termes de forçage.

- La gravité est notée g.
- La profondeur de l'océan est notée h.
  On néglige l'impact de l'élévation dans le bilan de masse.
  Par contre, il faudra bien tenir compte de la bathymétrie !
- Tous les effets dissipatifs (effets visqueux, effets des tourbillons de petites échelles, frottement exercé par le fond du bassin) sont modélisés par un terme proportionnel à la vitesse : une telle approximation est évidemment discutable ! Elle a toutefois le mérite de la simplicité et de fournir des prédictions acceptables. Ce facteur de proportionnalité est noté γ.
- Le terme de Coriolis fait intervenir un facteur $f = 2\Omega \sin\theta$ où Ω et θ sont respectivement la vitesse de rotation de la terre et la latitude.
- On imposera une condition initiale en élévation : c'est le déplacement d'une partie de la mer qui va générer le tsunami. On supposera -par contre- que l'océan est au repos : ce qui n'est pas totalement exact, puisqu'il y a les marées et les courants océaniques globaux qu'on supposera donc ici négligeables par rapport à l'effet du tsunami...

Les paramètres à utiliser dans les simulations sont :

$\Omega = 2\pi$ / 86 400	$\gamma = 10^{-7}$
$g = 9.81$	$R = 6\ 371\ 220$

1.3 La méthode numérique...

Il s'agit de résoudre les équations -ci dessus- en utilisant la méthode de Galerkin discontinue avec des fonctions de forme de degré un. Pour l'intégration temporelle, on fera usage de la méthode d'Euler explicite. L'intégration numérique des diverses intégrales se fera avec une méthode d'Hammer à 3 points pour les triangles et une méthode de Gauss-Legendre à deux points pour les arêtes? Les intégrales se font sur le plan stéréographique.

On peut montrer que les équations discrètes à résoudre sont :

$$\begin{aligned}
\sum_{j=1}^{n} < \phi_i\phi_j >_e \frac{dH_j^e}{dt} &= < \left(\frac{\partial\phi_i}{\partial x} h u_e^h + \frac{\partial\phi_i}{\partial y} h v_e^h\right)\left(\frac{4R^2+x^2+y^2}{4R^2}\right) >_e \\
&\quad + < \phi_i \left(\frac{h(xu+yv)}{R^2}\right) >_e + \ll \phi_i h u_n^* \left(\frac{4R^2+x^2+y^2}{4R^2}\right) \gg_e \qquad i = 1,\ldots,n \\
\sum_{j=1}^{n} < \phi_i\phi_j >_e \frac{dU_j^e}{dt} &= < \phi_i\big(f v_e^h - \gamma u_e^h\big) + \frac{\partial\phi_i}{\partial x} g\eta_e^h \left(\frac{4R^2+x^2+y^2}{4R^2}\right) >_e \\
&\quad + < \phi_i \left(\frac{gx\eta}{2R^2}\right) >_e + \ll \phi_i n_x g\eta^* \left(\frac{4R^2+x^2+y^2}{4R^2}\right) \gg_e \qquad i = 1,\ldots,n \\
\sum_{j=1}^{n} < \phi_i\phi_j >_e \frac{dV_j^e}{dt} &= < \phi_i\big(-f u_e^h - \gamma v_e^h\big) + \frac{\partial\phi_i}{\partial y} g\eta_e^h \left(\frac{4R^2+x^2+y^2}{4R^2}\right) >_e \\
&\quad + < \phi_i \left(\frac{gy\eta}{2R^2}\right) >_e + \ll \phi_i n_y g\eta^* \left(\frac{4R^2+x^2+y^2}{4R^2}\right) \gg_e \qquad i = 1,\ldots,n
\end{aligned}$$

avec la vitesse normale sur un segment définie par $u_n = (un_x + vn_y)$. Le vecteur $\mathbf{n} = (n_x, n_y)$ est le vecteur normal unitaire sortant pour l'élément Ω_e. Afin d'obtenir une solution numérique stable et précise, il faut évaluer avec subtilité les valeurs des flux sur les segments. En définissant les valeurs à gauche (le premier élément :-) et à droite (le second élément :-) pour un segment par η^L, u_n^L et η^R, u_n^R, on calculera les valeurs des flux sur base des expressions suivantes :

$$\begin{aligned}
\eta^* &= \left(\frac{\eta^L + \eta^R}{2}\right) + \sqrt{\frac{h}{g}}\left(\frac{u_n^L - u_n^R}{2}\right) \\
u_n^* &= \left(\frac{u_n^L + u_n^R}{2}\right) + \sqrt{\frac{g}{h}}\left(\frac{\eta^L - \eta^R}{2}\right)
\end{aligned}$$

Ces deux expressions[1] sont obtenues sur base du calcul des courbes caractéristiques du problème unidimensionnel défini selon un axe perpendiculaire au segment, tel qu'illustré dans le cours sur le tsunami unidimensionnel...

[1] Retrouver ces deux expressions a été fait pendant le cours :-)

Pour les segments frontières, on imposera une condition d'imperméabilité des frontières et une condition naturelle pour l'élévation en imposant que $\eta^R = \eta^L$ et $u_n^R = -u_n^L$. La seconde relation consiste à exiger de manière faible que la vitesse normale est nulle à la frontière.

Pour obtenir une solution, on ne peut accepter une bathymétrie nulle :-) : nous avons donc systématiquement changer toutes la valeurs inférieures à 100 mètres à cette valeur. Cette valeur est un relatif bon compromis entre précision et stabilité, quoique choisie de manière un peu arbitraire. Afin de pouvoir évaluer et comparer vos projets, il vous est demandé de ne modifier en aucun cas ces valeurs de bathymétrie et de toujours considérer une bathymétrie linéaire. Il est toutefois possible de discuter de cette valeur dans votre rapport :-)

Pour la condition initiale, il faut impérativement effectuer une interpolation aux valeurs nodales pour obtenir la solution de référence. Effectuer un lissage est donc proscrit, même si cela n'aurait pas été une mauvaise idée : il vous est possible d'en discuter dans votre rapport si cela vous semble opportun.

Quatre maillages en coordonnées stéréographiques avec la bathymétrie et la condition initiale sont fournis par nos soins. Pour chaque sommet, nous vous fournissons les deux coordonnées stéréographiques et la bathymétrie. On vous fournit également une fonction permettant de calculer la valeur initiale de l'élévation qui va générer le tsunami d'Okada.

1.4 Ce que vous devrez réaliser !

L'objet du projet est de réaliser un petit code d'éléments finis permettant de prédire le tsunami : il s'agit d'un petit laboratoire numérique où vous allez être confrontés à des comportements numériques un petit peu inhabituels et où il s'agira de les interpréter correctement : il y a peu à programmer et beaucoup à comprendre ! Votre projet consistera à réaliser deux programmes.

1. Le premier programme contiendra la résolution numérique du problèmes sans aucun graphique avec l'implémentation de la fonction suivante :

   ```
   [U,V,E] = compute(theMeshFile,theResultFiles,U,V,E,dt,nIter,nSave)
   ```

   où on spécifie le pas de temps, le nombre d'itérations à effectuer, le nombre d'itérations entre chaque sauvegarde de résultats dans un fichier. On donne également le nom des fichiers pour la lecture du maillage et le format pour écrire les divers fichiers de résultats. Finalement, on fournit les valeurs nodales des conditions initiales sous la forme de tableaux de taille $n \times 3$ où n est le nombre de triangles du maillage. La fonction renvoie les valeurs nodales finales obtenues à l'issue de toutes les itérations : on ne conserve donc pas les valeurs intermédiaires sauf celles qui sont sauvegardées dans les fichiers de résultats.

   L'appel de `compute("PacificTriangleFine.txt","tsunami-%06d.txt",U,V,E,0.1,400,100)` devrait donc générer la création de 4 fichiers qui s'appelleront respectivement :

   ```
   tsunami-000100.txt
   tsunami-000200.txt
   tsunami-000300.txt
   tsunami-000400.txt
   ```

   contenant l'ensemble des valeurs nodales d'élévation. Les 3 valeurs nodales d'élévation de chaque élément seront données sur une ligne distincte. Un petit programme de test vous est fourni pour tester votre programme : il contient aussi les fonctions permettant d'écrire et de lire les fichiers de résultats.

2. Pour tester votre implémentation, le programme `tsunamiTest.py` est disponible :

```
import numpy as np
import tsunami as tsunami

theMeshFile = "PacificTriangleFine.txt"
[nNode,X,Y,H,nElem,elem] = tsunami.readMesh(theMeshFile)
print(" == Number of elements : %d " % nElem)
print(" == Number of nodes    : %d " % nNode)

x = np.zeros([nElem,3])
y = np.zeros([nElem,3])
for iElem in range(nElem):
  nodes  = elem[iElem]
  x[iElem][:] = X[nodes]
  y[iElem][:] = Y[nodes]
E = tsunami.initialConditionOkada(x,y)

theResultFiles = "eta-%06d.txt"
tsunami.writeResult(theResultFiles,0,E)

U = np.zeros([nElem,3])
V = np.zeros([nElem,3])
E = tsunami.readResult(theResultFiles,0,nElem)
dt = 0.1; nIter = 100; nSave = 25
[U,V,E] = tsunami.compute(theMeshFile,theResultFiles,U,V,E,dt,nIter,nSave)

for iElem in [27,28] :
  print(" == Elevations for element %d : %14.7e %14.7e %14.7e " % (iElem,*E[iElem][:]) )
```

3. En outre, on a fourni quelques fichiers de résultats qu'a produit l'enseignant.... Cela devrait vous donner une idée de ce que vous devriez obtenir comme résultats. Attention : ce sont des résultats plausibles, mais ce ne sont pas les résultats exacts de la simulation : il ne faut donc pas prendre les valeurs fournies comme des valeurs de référence à reproduire absolument. Pour être vraiment tout-à-fait honnête, ils sont même légèrement biaisés à dessein pour vous éviter toute tentation :-)

4. Le second fichier à remettre sera une version modifiée `tsunamiAnimate.py` et devra contenir l'entièreté de l'implémentation graphique permettant la visualisation de votre tsunami en améliorant le code fourni. Cette seconde partie est optionnelle et ne sera pas testée sur le serveur, mais de manière manuelle par les assistants : il vous sera loisible de faire une démonstration de votre programme lors de l'interview orale du projet. Il est possible de remettre le fichier `tsunamiAnimate.py`, mais si vous n'avez effectué aucune modification de ce dernier : il sera difficile de vous reconnaître une contribution importante :-) Si des packages particuliers ou une version particulière de `python` est requise pour faire fonctionner votre code, il est utile de l'indiquer clairement dans les commentaires et dans votre rapport. Pour ce second fichier, il est utile et même largement recommandé d'inclure des commentaires pour que le correcteur puisse apprécier à sa juste valeur votre subtile implémentation.

En conclusion, il vous est demandé :

1. De concevoir un programme permettant la simulation du tsunami avec des fonctions de forme linéaires discontinues.

2. D'optimiser votre programme afin qu'il soit le plus rapide et le plus joli possible.

3. De rédiger une note de synthèse d'au maximum 4 pages pour le Service d'Océanographie en expliquant comment obtenir les équations des eaux peu profondes pour les coordonnées stéréographiques. Fournir une estimation de l'ordre de précision du résultat obtenu en expliquant comment vous avez

validé votre code numérique. Produire quelques illustrations pertinentes pour l'analyse de la solution. Expliquer comment vous avez optimisé votre programme afin qu'il soit le plus rapide possible. Ne pas recopier les développements théoriques du syllabus, ne pas recopier l'énoncé du problème, ne pas fournir des diagrammes incompréhensibles, ne pas donner des tableaux de chiffres indigestes. L'orthographe, le soin et la présentation seront conformes à celles d'une note fournie par un bureau d'études professionnel.

(option) Améliorer le code graphique fourni afin de réaliser une animation mettant en scène efficacement le tsunami sur base des résultats obtenus et stockés dans les fichiers.

L'entièreté de votre code de calcul sera inclus dans un unique fichier `tsunami.py` tandis que l'implémentation graphique sera fournie dans un unique fichier `tsunamiAnimate.py`. Il vous est loisible de reprendre tout ce que vous souhaitez dans les codes fournis pour 7 premiers devoirs. Uniquement, la partie non-graphique devra pouvoir être exécutée sur le serveur. Lors de la soumission, un calcul d'un nombre limité de pas de temps sera effectué afin de vous permettre d'estimer la rapidité de votre programme.

Toutes les soumissions seront soumises à un logiciel anti-plagiat. En cas de fraude flagrante, les cas de plagiat seront soumis au Jury des examens. Vous êtes invités à consulter la page web de l'Université pour avoir une petite idée des sanctions possibles dans ce cas !

1.5 Evaluation du projet

L'évaluation finale du projet se fera sur base d'une interview où vous serez invité à faire une démonstration de vos deux programmes sur votre ordinateur (en particulier, l'implémentation graphique :-). Les interviews se feront les vendredi 17 mai, lundi 20 mai et mardi 21 mai 2019. L'interview comprendra également des questions sur la compréhension théorique de la matière : les groupes qui auront réussi de manière brillante le projet et l'interview pourraient être éventuellement dispensés de l'examen final.

L'échéance pour la remise du projet a été repoussée à la date du lundi 6 mai 2019 à 23h59 pour tenir compte du retard accumulé dans les premiers devoirs et la remise plus tardive de l'énoncé que prévu. Nous avons aussi réduit l'ampleur du projet afin que celui ne prenne pas trop de temps :-) Bon courage à tous !

L'évaluation du projet se fera sur la base suivante sur un total de 20 points :

Programme linéaire qui fonctionne	5
Précision des résultats	5
Rapidité du code de calcul	5
Interview et rapport	5
Implémentation graphique	5

Comme vous pourrez le constater, il est parfaitement possible d'obtenir 20/20 sans réaliser l'implémentation graphique. C'est donc bien une option qu'il n'est pas indispensable de réaliser.

1.6 Grand Prix International 2019 de l'Elément le Plus Fini

Le programme correct le plus rapide recevra le Grand Prix de l'Elément le Plus Fini d'un montant de 100 euros. Ce prix a été rendu possible grâce à la contribution anonyme d'un ancien étudiant soucieux de

promouvoir la qualité de la formation. La proclamation des résultats se fera sur le web avant le 15 juin 2019. Les gagnants du prix seront invités à prendre contact avec le titulaire du cours. La mesure de la rapidité du programme sera effectuée par deux tests indépendants avec le maillage le plus raffiné et des éléments cubiques. Le résultat final sera la moyenne des deux mesures. En cas de timings aberrants ou manifestement parasites, des runs complémentaires seront effectués. Le règlement du concours n'a pas été rédigé et n'est donc pas disponible auprès d'un huissier de justice à Ottignies-Louvain-la-Neuve.

Les auteurs de l'implémentation informatique graphique la plus remarquable gagneront le prix spécial des assistants d'un montant de 50 euros. Cette seconde partie du concours est également ouverte à l'équipe didactique et à tous les occupants du bâtiment Euler.