RF

Manuel de Poutrix 0.0.3

Introduction

Poutrix a pour ambition de devenir un logiciel de résolution numérique de problèmes physiques unidimensionnels aux dérivées partielles. Je me suis concentré ici à donner au programme un squelette comprenant l'import des résultats et leur export, en me restreignant aux équations de poutres dans les hypothèses d'Euler-Bernoulli.

Poutrix 0.0.3 est une version beta de la version 0.1. Elle permet de résoudre des problèmes statiques et dynamiques sur poutres quelconques et en matériaux isotropes, avec des éléments d'ordre 1 uniquement. Un solveur de problèmes en grands déplacements est disponible, bien que relativement limité pour l'instant.

Tandis que les parties dynamiques ne sont pour l'instant disponibles qu'à titre de démonstration, le solveur statique est désormais utilisable pour résoudre les problèmes les plus classiques de RDM.

Table des matières

1 Utilisation du code **2**

1.1 Prérequis . . . 2

1.2 Fonctionnement . . . 2

1.3 Performances . . . 3

1.4 Fichier d'entrée . . . 3

1.4.1 Considérations générales sur la syntaxe . . . 3

1.4.2 Les poutres . . . 3

1.4.3 Les liaisons . . . 4

1.4.4 Le solveur . . . 4

1.4.5 Liste des affectations facultatives . . . 4

1.5 Fichier donnée annexe . . . 5

1.6 Utilisation des paramètres . . . 5

2 Liste des changements par rapport à Poutrix 0.0.2 **5**

2.1 Interface . . . 5

2.2 Syntaxe . . . 5

2.3 Solveur . . . 5

2.4 Post-traitement . . . 6

2.5 Modèle . . . 6

2.6 Organisation . . . 6

3 Erreurs possibles **6**

3.1 Retranscription des erreurs 6

3.2 Liste des erreurs signalées 6

1 Utilisation du code

1.1 Prérequis

Poutrix 0.0.3 nécessite Python 3.3.3, et ne fonctionne que sur Windows.

Les librairies python utilisées sont les suivantes : numpy, scipy, tkinter, math et os. La librairie sp est jointe avec le code.

On peut obtenir Python 3 et toutes ces librairies pour Windows en téléchargeant par exemple WinPython, qui comprend de plus Spyder, une interface python.

1.2 Fonctionnement

Dans la présente version du programme, la seule manière d'entrer des données pour le calcul est d'écrire un fichier assemblage au format texte qui suit la syntaxe imposée. Des fichiers d'exemple sont proposés. Un fichier .xml permettant de configurer la coloration syntaxique sous Notepad++ est joint avec le code.

On peut sélectionner quel fichier sera traduit par Poutrix au moyen d'une interface graphique, en cliquant sur le nom du fichier à ouvrir. Tous les fichiers présents dans le dossier où se trouve le code Python sont proposés à l'interprétation, y compris ceux qui ne sont pas lisibles. L'interface graphique se lance si on lance "lancer_pb.py". Cette interface subira encore des modifications dans les versions suivantes.

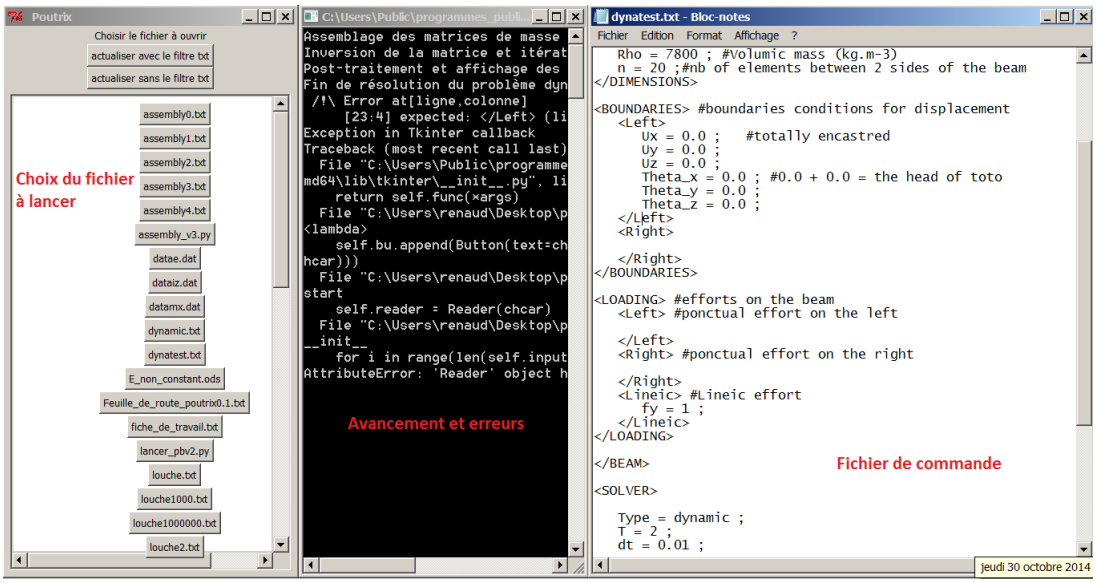

FIGURE 1 – Interface de Poutrix

On peut imposer un maillage pour une poutre donnée au travers d'un fichier .dat annexe, dans le cas contraire, le mailleur automatique intégré à poutrix divisera de façon régulière chaque partie de poutre entre deux liaisons (ou bords) en un nombre réglable d'éléments.

Poutrix sort un fichier texte contenant les valeurs du déplacement et des efforts aux différents noeuds. L'utilisateur choisit le contenu de ce fichier dans le fichier de commande.

1.3 Performances

Poutrix utilise la bibliothèque d'algèbre linéaire de numpy, qui est très largement compilée. Cependant, les opérations d'assemblage des matrices et de post-traitement sont interprétées. Il en résulte que les performances ne sont pas comparables avec un solveur codé en Fortran ou en C++. Un nombre de noeuds de l'ordre de quelques milliers (càd quelques dizaines de milliers de ddl) représente déjà un problème conséquent pour Poutrix (mais ceci dépend évidemment des performances de la machine).

Compte tenu de la précision numérique des flottants Python, il existe une taille minimale de maille telle que les résultats soient fiables. Cette taille est de l'ordre de $10^{-9}L$ avec L la longueur de la poutre. Des noeuds trop rapprochés seront considérés comme confondus par poutrix.

1.4 Fichier d'entrée

Le fichier d'entrée se divise en trois parties : tout d'abord les déclarations des poutres, avec leurs paramètres géométriques et matériaux, puis les déclarations des liaisons, et enfin des informations sur la résolution souhaitée. Ce fichier est lu par un parser utilisant la bibliothème sp.

1.4.1 Considérations générales sur la syntaxe

Sauf indication contraire, les espaces et les sauts de ligne ne sont pas interprétés par Poutrix, ce qui signifie que l'on peut s'en servir pour organiser son fichier de commande.

Les commentaires doivent être précédés du symbole #, et peuvent être placés n'importe où sur une ligne. Aucun caractère placé sur une ligne après un # ne sera interprété.

Il est possible, lors de l'affectation de certaines valeurs sur l'ensemble de la poutre, d'écrire en argument non pas une valeur, mais le nom d'un fichier de données qui contient une liste de valeurs. Ce faisant, l'utilisateur impose aussi un maillage pour la poutre. Il est question de ceci dans la section "Fichier annexe de données".

De plus, il est possible d'utiliser des paramètres : au lieu d'écrire une valeur, on peut écrire une expression qui contient des paramètres donnés dans la section "paramètres" et éventuellement X pour l'abscisse de la poutre. Ceci permet par exemple d'avoir une poutre à section variable, ou un chargement non uniforme sur la poutre. Voir la section "Utilisation des paramètres".

L'ordre d'entrée des données est important et ne peut pas être perturbé sous peine d'erreur fatale pour le programme.

1.4.2 Les poutres

La déclaration d'une poutre commence par une balise de début de poutre, et finit par la balise de fin de poutre. Elle se divise elle-même en trois parties : la déclaration des caractéristiques (géométriques et matériaux), des conditions limites aux deux bords de la poutre (ddl imposé ou pas, nul ou non), et enfin les chargements mécaniques. Les chargements doivent être donnés dans le repère global.

On peut utiliser la commande "defo$_{i}nit$"(ou"$init_{d}efo$")$pourdonnerà lapoutreunedéformationinitialedépendan

1.4.3 Les liaisons

La déclaration d'une liaison comporte six affectations : le nom de la première poutre, l'abscisse curviligne du point de liaison sur la première poutre, le nom de la seconde poutre, l'abscisse curviligne du point de liaison sur la deuxième poutre, l'axe ou la normale de liaison (facultatif) et enfin le repère dans lequel est donnée la liaison (facultatif).

L'ordre de déclaration des poutres n'a aucune incidence sur le traitement de la liaison. Pour définir une liaison d'une poutre avec le bâti, il faut renseigner la partie "boundaries" de la poutre, et non pas définir une nouvelle liaison dans la partie "Links".

Voici la liste des mots-clef correspondant à des liaisons : rot (rotule), enca (encastrement), splan (sphère-plan), aplan (appui-plan), pivot, glis (glissière), pglis (pivot-glissant), scyl (sphère-cylindre), sdoi (sphérique à doigt). Il est à noter que les liaisons cylindre-plan et hélicoïdale manquent.

Remarque : il est possible dans Poutrix de lier deux points qui ne sont pas coïncidents, leurs déplacements seront alors égaux, mais ceci ne les rendra pas coïncidents pour autant.

1.4.4 Le solveur

Dans la partie solveur, on effectue deux affectations et on place les blocs de post-traitement. En premier, on donne la nature du problème : statique, modal ou dynamique. Dans le cas d'un problème modal, on peut placer un nombre entier après le mot modal pour indiquer le nombre de modes voulus. Dans le cas d'un problème dynamique, il faut ajouter deux lignes donnant respectivement la durée de l'étude et la période d'échantillonage souhaitées.

Pour ce qui est la résolution des problèmes dynamiques, dans la version actuelle, les efforts imposés ne peuvent pas varier dans le temps, c'est à dire qu'on ne peut imposer que des échelons de Heaviside. Ceci fera l'objet de modifications. Par ailleurs, les coefficients du schéma de Newmark utilisé sont $\beta = \frac{1}{4}$, $\gamma = \frac{1}{2}$, et ne peuvent pas être modifiés. En particulier, rien n'a encore été prévu pour effectuer des résolutions explicites.

Dans le cas d'une résolution statique en grands déplacements, la méthode utilisée est une méthode itérative de type Euler, et dont la convergence n'est donc pas assurée. Le nombre d'itérations peut être rajouté en une ligne : n_iter =.

On donne ensuite le nom du fichier de sortie (sans l'extension : le programme rajoutera .txt).

Enfin, il faut ajouter des informations sur la sortie voulue : on peut placer un nombre quelconque de blocs sortie qui s'organisent chacun de la façon suivante : le nom de la poutre, les composantes du vecteur des déplacements généralisés que l'on veut voir apparaître, et enfin les composantes du vecteur des efforts par éléments et par noeuds.

On peut de plus, dans le cas d'un comportement dépendant du temps, placer des blocs de sortie temporelle : ils permettent d'afficher la valeur d'une composante du vecteur des déplacements généralisés en un point donné en fonction du temps.

Il existe enfin des blocs de sortie destinés à l'affichage. Ils permettent d'afficher sous forme tabulaire, avec des séparateurs choisis par l'utilisateur, les coordonnées des points après déformation, ainsi que les déplacements et les efforts.

1.4.5 Liste des affectations facultatives

Certaines données sont facultatives et sont considérées comme nulles si elles ne sont pas données : c'est le cas des chargements ponctuels et linéiques.

Pour ce qui est des ddl imposés, le fait de ne pas les exprimer conduit Poutrix à considérer que l'on a affaire à un bord libre.

En ce qui concerne le nombre de modes propres que l'on veut visualiser, ne rien mettre conduit à considérer ce nombre égal à 1.

Dans la partie solveur, on peut rajouter, dans le cas d'un problème dépendant du temps, la durée de l'étude et la période d'échantillonnage (entre le type de solveur et le nom du fichier de sortie). Si celles-ci ne sont pas données, elles seront arbitrairement prises égales respectivement à 1 et $0,1$.

1.5 Fichier donnée annexe

Un fichier annexe de données donne une liste de valeurs nodales en remplacement d'une seule valeur. Pour affecter un fichier annexe, l'utilisateur devra écrire son titre. Les fichiers de données doivent porter l'extension .dat. Par exemple, si on veut appeler "toto.dat", pour un module d'Young variable, il faut taper : "E = toto".

Les valeurs données dans un fichier annexe le sont aux noeuds du maillage considéré, et ce de la façon suivante : le fichier doit alterner un réel, donnant la valeur au noeud, et un entier donnant la longueur relative entre ce noeud et le suivant.

Ce format n'est pas destiné à rester tel quel et subira probablement des modifications dans les versions ultérieures de Poutrix.

1.6 Utilisation des paramètres

L'utilisation de paramètres et d'expressions mathématiques dépendant éventuellement de l'abscisse dans les affectations est destinée à remplir une bonne partie des fonctionnalités auparavant assurées par les fichiers de données annexes, moins pratiques d'emploi.

Pour l'instant, il existe une limitation : les formules ne doivent pas comprendre d'espace. Par exemple : "E = E0 + (E1-E0)/L*X", qui vise à obtenir un module d'Young variant linéairement, conduira à une erreur, et il faut taper : "E = E0+(E1-E0)/L*X". De plus, tous les paramètres utilisés doivent être affectés dans la partie "Paramètres".

2 Liste des changements par rapport à Poutrix 0.0.2

2.1 Interface

*Possibilité de naviguer dans les sous-répertoires.

2.2 Syntaxe

*Introduction d'une syntaxe française

*Possibilité d'utiliser des paramètres

*Possibilité de donner une déformée initiale

*Possibilité d'imposer des chargements dépendants de l'espace

*Syntaxe allégée avec plus de zones facultatives

*Coloration syntaxique pour Notepad++

2.3 Solveur

*Optimisation de l'assemblages

2.4 Post-traitement

*Possibilité de sortir les coordonnées globales de tous les points

*Torseur de cohésion au bord des éléments

2.5 Modèle

*Poutres courbes

*Grands déplacements (pas fini)

*Possibilité d'introduire un repère local

*Liaisons selon les repères locaux

2.6 Organisation

*Le changement de base se fait dans l'élément

3 Erreurs possibles

3.1 Retranscription des erreurs

Lors du lancement de "lancer_pb.py", en plus de la fenêtre de choix, une fenêtre apparaît dans laquelle sont listées les erreurs python d'exécution. Si l'erreur est une erreur de syntaxe poutrix, elle apparaît dans les premières lignes sous la forme suivante : "/!\ Error at [ligne,colonne]". C'est cette erreur qui donne ce que le parser attendait dans le fichier de commande, et qu'il n'a pas trouvé.

3.2 Liste des erreurs signalées

*Affichage : L'interface est encore un peu un bug à elle toute seule.

*Solveur : la résolution des problèmes de modes propres ne converge pas (probablement est-ce dû à des erreurs numériques).

*Modélisation : il est impossible d'imposer l'orientation angulaire de la poutre autour de son axe.

*Et sûrement bien d'autres encore, qui n'attendent que d'être signalées...