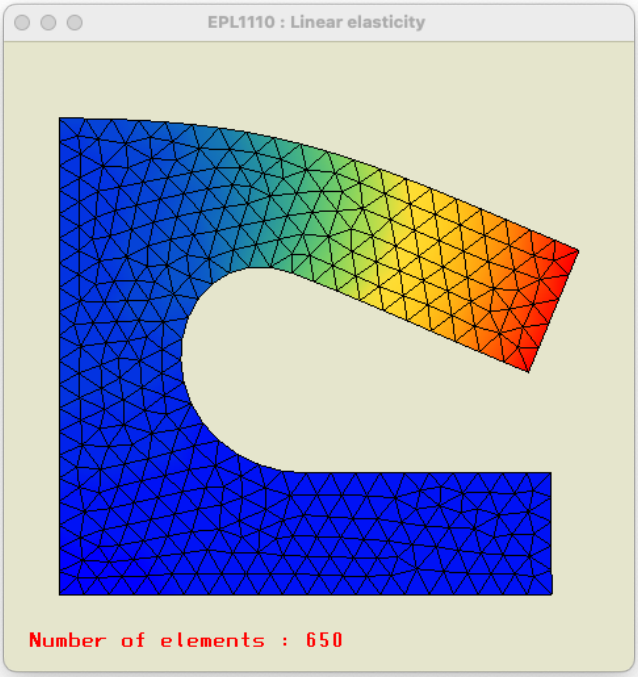
Nathan Tihon, Miguel De Le Court, Alexandre Chemin, Gatien Dony, Michel Henry, Antoine Quiriny, Florian De Bel, Vincent Degrooff, Vincent Legat, Thomas Leyssens, Simon Yans

1 Elasticité linéaire et 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 C pour l'élasticité linéaire.

Objectifs du projet...

L'objectif du projet sera triple cette année !

1. La toute première étape consistera à imaginer un problème d'élasticité linéraire plane ou axisymétrique à résoudre. Cela consiste à définir une géométrie, à sélectionner des paramètres matériels et à définir les conditions aux frontières en termes de déplacements ou de forces de contact. On sera attentif à bien veiller à ce que les conditions aux limites n'autorisent pas un déplacement rigide : ce qui rendrait le problème mathématique mal posé. Il sera aussi fondamental de déterminer quels sont les valeurs numériques utiles à obtenir lors de la simulation. Est-ce que les tensions dépassent une valeur limite maximale, est-ce que les déformations deviennent inacceptables ou est-ce qu'il s'agit d'identifier une géométrie optimale. Et surtout, nous convaincre que simuler ce problème a un réel intérêt pratique ! Regardez aussi le monde qui vous entoure et repérez toutes les pièces métalliques ou structures de la vie de tous les jours : support d'étagère, tuyauteries diverses,

diapason, ponts, voutes de cathédrale ou que sais-je encore.... Ne soyez pas trop ambitieux : la simplicité est aussi la clé du succès.

2. Les conditions frontières de votre problème devront pouvoir s'écrire exclusivement en composantes x et y. Il faut donc veiller à définir un problème qu'il est possible de résoudre sans faire usage de conditions en composantes normales et tangentielles.

3. Ensuite, il vous sera demandé d'écrire un code informatique efficace, bien documenté et robuste pour résoudre des problèmes d'élasticité linéaire en deux dimensions. Votre programme devra pouvoir résoudre des problèmes en tensions ou en déformations planes ou des problèmes axisymétriques. Il devra pouvoir fonctionner avec des éléments triangulaires linéaires (et éventuellement avec desquadrilatères bilinéaires). En outre, vous devrez être capables de retrouver les composantes du champ de contraintes aux noeuds du maillage. Pour que votre programme soit efficace, il faut utiliser soit un solveur bande, soit un solveur frontal, soit un solveur itératif tel que les gradiens conjugés.

4. Finalement, vous utiliserez votre code pour résoudre le problème que vous avez choisi, vous analyserez vos résultats.

Evidemment, les devoirs `GeoMesh`, `Band` ,`LinearElasticity` et `LinearElasticityForces` contiennent beaucoup d'indices pour réaliser votre projet. Dans la solution de ce dernier devoir, vous avez déjà la résolution des équations de l'élasticité linéaires pour des triangles linéaires et des quadrilatères bilinéaires avec un solveur plein... Il ne reste donc qu'à généraliser et à optimiser ce qui a été réalisé dans ce devoir

Plusieurs séances d'exercices seront aussi consacrés à la présentation des aspects plus techniques du projet et serviront de séances globlales de questions-réponses. Il sera évidemment possible de chercher de l'aide, de l'assistance auprès de avec votre assistant et votre tuteur de référence qui vous accompagneront pendant tout le projet. C'est votre responsabilité de prendre contact avec eux et d'organiser votre projet. N'hésitez pas non plus à utiliser les canaux Teams mis en place avec eux.
Il faut commencer à faire votre projet le plus rapidement possible !
On verra pendant le cours tout ce qui est indispensable pour faire le projet.

2 Ce qu'il s'agit de faire !

2.1 Définir votre problème d'élasticité linéaire !

Vous rédigerez une petite note d'une unique page maximum à soumettre à votre assistant et votre tuteur de référence qui devront l'approuver. Cette note doit contenir la description géométrique du problème, une image d'un maillage caractéristique généré avec `gmsh` et la description complète des conditions aux frontières. Il devra être approuvé par votre assistant de référence avant le vendredi 28 février : Oui : c'est quasiment demain :-).

Cette première étape est la partie réellement créative du projet : il faut trouver, imaginer, découvrir un problème pour lequel vous allez utiliser votre programme ! Evidemment, vous pourriez vous contenter de résoudre un exemple fourni par les assistants, mais ce n'est ni très excitant, ni très original ! On vous demande donc de nous proposer un problème que vous allez résoudre :-)

Evidemment, si votre problème fait intervenir de la transition, de l'innovation ou de la diversité : c'est encore mieux ! La transition, c'est faire des éléments finis recyclables, durables et verts, l'innovation c'est faire de nouvelles choses et la diversité, c'est s'ouvrir à tous les éléments finis présents dans notre sympathique communauté universitaire :-)
Là, aussi c'est le défi qu'il vous appartient de relever avec brio !

2.2 Ecrire un code d'éléments finis !

On vous demande d'écrire un code C d'éléments finis qui résoud de manière efficace un problème d'élasticité linéaire plane. Ce code n'aura aucune option graphique et ne pourra faire appel à aucune librairie externe : en particulier, vous ne pouvez pas utiliser la libraire de `gmsh` ou des solveurs itératifs tels que `petsc` ou d'autres choses. La totalité de l'implémentation doit être faite par vos soins : il est permis de vous inspirer partiellement du contenu `fem.c` et `fem.h` pour votre code, mais il est requis de bien isoler cela dans des fichiers distincts. En cas de doute, n'hésitez pas à consulter votre assistant(e) de référence. On utilisera une règle d'intégration à trois points sur les triangles, à quatre points sur les quadrilatères telles que définis dans le dernier devoir. Pour les intégrales sur les segments frontières, on utilisera une règle de Gauss-Legendre usuelle à deux points.

Le code sera soumis sous la forme d'un premier fichier

`group001-vlegat-jfremacle.zip`

contenant l'ensemble des sources du programme, un fichier `CMakeLists.txt` et un fichier `ReadMe.txt` contenant les instructions détaillées pour faire fonctionner le code. L'arborescence des fichiers correspondra à celle des devoirs et le nom du répertoire respectera exactement la forme suivante :

```
group001-vlegat-jfremacle\build
group001-vlegat-jfremacle\CMakeLists.txt
group001-vlegat-jfremacle\ReadMe.txt
group001-vlegat-jfremacle\data
group001-vlegat-jfremacle\src
```

Il faut évidemment adapter le numéro de groupe et les deux identifiants à votre cas particulier !

Le programme acceptera deux arguments en ligne de commande pour spécifier les noms de deux fichiers. Par défaut, le programme ouvrira un fichier `data/mesh.txt` contenant le maillage et un fichier `data/problem.txt` contenant la description du problème. Les fonctions pour écrire et lire le maillage sont disponibles dans le dernier devoir et un exemple est également fourni pour le fichier décrivant le problème.

Le programme fournira les résultats obtenus dans les fichiers indiqués par le fichier `data/problem.txt`.

2.3 Résoudre et analyser votre problème particulier !

Il s'agit donc bien de créer une géométrie comme vous l'avez fait dans le devoir `GeoMesh` et puis de visualiser vos résultats de manière originale et efficace.... Il s'agit donc d'écrire d'autres petits programmes qui effectuent ces tâches de pré-processing et de post-processing. Toutefois même si on vous demande de nous fournir les sources des programmes réalisés : ils ne seront ni analysés, ni corrigés.... Seul, le résultat final obtenu est important ici ! Réaliser une animation de votre simulation est une option possible, ou même effectuer un test expérimental pour estimer les ordres de grandeur est aussi une option possible. Evidemment, tout sera considéré comme d'éventuels bonus.

Les données pour le problème analysé seront fourni dans un second fichier

`group001-vlegat-jfremacle-rapport.zip`

avec l'arborescence suivante :

```
group001-vlegat-jfremacle-rapport\build
```

```
group001-vlegat-jfremacle-rapport\CMakeLists.txt
group001-vlegat-jfremacle-rapport\ReadMe.txt
group001-vlegat-jfremacle-rapport\data\mesh.txt
group001-vlegat-jfremacle-rapport\data\problem.txt
group001-vlegat-jfremacle-rapport\src
```

Il faut y inclure les sources pour compiler les programmes éventuels de post-processing et de pre-processing que vous avez réalisés. Y joindre les fichiers de données du problème analysés avec obligatoirement au moins un fichier `mesh.txt` et un fichier `problem.txt` Si vous avez utilisé `github` pour écrire votre code et/ou votre rapport, vous pouvez l'indiquer dans le fichier `ReadMe.txt`. Il est permis d'éventuellement fournir ce dernier fichier sous le format `ReadMe.md`, si cela correspond davantage à votre manière de travailler.

2.4 Et concrètement ?

Plus précisément, il vous est demandé :

1. De définir un problème original à résoudre.
2. De concevoir un programme d'élements finis pour l'élasticité linéaire avec des triangles linéaires.
3. D'implémenter les modèles de déformations planes et de tensions planes.
4. D'implémenter l'élasticité linéaire axisymétrique.
5. (option) D'implémenter des quadrilatères bilinéaires.
6. (option) D'implémenter des conditions frontières en termes de composantes normale et tangentielle.
7. (option) D'implémenter un solveur linéaire frontal.
8. (option) D'implémenter un algorithme de renumération des noeuds ou des éléments.
9. (option) De calculer les valeurs des composantes de tension aux noeuds.
10. (option) De réaliser une animation de votre simulation et du problème analysé.
11. (option) D'effectuer une analyse expérimentale du problème analysé.

2.5 Evaluation du projet

L'évaluation du projet se fera sur la base suivante sur un total de 30 points :

Programme d'éléments finis /10	
Respect strict du format de soumission (nom des fichiers !)	2
Programme qui fonctionne et fournit un résultat qui semble cohérent	8
(bonus) Implémentation d'options supplémentaires	4
Problème analysé /10	
Définition du problème : analyse physique et originalité	4
Analyse et validation des résultats	4
Pertinence de l'analyse	2
(bonus) Animations et esprit critique	4
Participation et interview /10	
Participation active aux projet et interaction avecl l'assistant de références	4
Interview et démonstration du projet	6

Comme vous pourrez le constater, il est parfaitement possible d'obtenir 30/30 sans réaliser aucun bonus.

Votre projet (programme, rapport, animation) sera soumis à votre assistant de référence selon les modalités qu'il vous indiquera en temps utile : cet assistant de référence testera votre code et vérifiera qu'il arrive bien à l'exécuter. Il est évidemment primordial de rencontrer et d'interagir avec votre assistant de référence avant la date de l'échéance du projet.

Pour rappel, 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 !

2.6 Assistant et tuteur de référence...

Est-ce que cela ressemble aux projets de l'années précédentes ?
Un peu mais on a essayé d'être différents !

En particulier, chaque groupe est vraiment invité à réaliser quelque chose de particulier et de différent des autres groupes, même si vous allez tous réaliser le même programme d'éléments finis. Ecrire le code n'est pas compliqué, mais avoir un rapport et un problème original est un vrai défi : et là, vous ne pourrez pas faire appel à vos condisciples car chaque groupe devra avoir un problème distinct. Pour vous aider dans cet aspect particulier, tous les groupes auront un assistant de référence qu'ils pourront contacter [1]

Votre assistant de référence sera également présent lors de votre interview : c'est votre allié dans ce projet, il peut vous aider en cas de problème informatique ou de difficultés particulières : c'est évidemment votre interlocuteur priviliégé pour le projet.

[1] En fonctions de modalités fournies par chaque assistant, soit lors des périodes de consultance qu'il aura définies, soit sur rendez-vous préalable. Toutes ces informations se trouveront sur les groupes Teams des assistants.

2.7 Grand Prix International 2025 de l'Elément le Plus Fini

Le meilleur projet (en termes de vitesse d'exécution) 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 2025. Les gagnants du prix seront invités à prendre contact avec le titulaire du cours. Les gagnants du prix acceptent que leur contributions soient publiées sur le site web de cours pour la postérité des éléments finis.

Cette année ouvre aussi une catégorie spéciale ouverte à tous, incluant tuteurs, assistants ou tout autre membre du batiment Euler. Les auteurs de l'implémentation la plus rapide de toutes gagneront le prix spécial des assistants d'un montant de 100 euros.

Attention cependant, le concours est purement optionnel. Il n'est donc pas indispensable de passer un temps monstrueux à celui-ci. Toutefois, cela peut titiller la curiosité de quelques étudiants qui trouvent que tout ceci est bien trop facile.

2.7.1 Détails pratiques

Pour la première fois en 2025, l'équipe didactique a décidé d'étendre le matériel autorisé au-delà d'un simple coeur. Tout processeur ou GPU branché à la machine de test sera utilisable! Pour accéder à ces ressources, les langages suivants sont autosisés :

- Le langage C et ses librairies standard (`pthreads`, `#pragma omp parallel for`, ...)
- Le langage C++ et toute la STL (`std::execution::unseq`, `std::experimental::simd`, ...)
- Cuda et ses librairies, à l'exception de cuBLAS, cuFFT, cuSPARSE, cuSOLVER (sinon le meilleur code tient en 2 lignes et ce n'est pas rigolo :-)
- OpenCL (3.0) et OpenGL (4.6)

L'utilisation de toute autre librairie externe est formellement interdite. Néanmoins le choix du standard C/C++ est libre (C99/C11/gnu99,...)

Les tests de performance s'effectuerons sur *Bobby*, le nouveau serveur MEMA. *Bobby* est équipé de :

- Un processeur Intel core i5-13500
- De 32GB de RAM
- D'une carte graphique NVIDIA RTX A4000 (compute capability 8.6)
- Les compilateurs disponibles sont: gcc-12 (g++-12), clang14 (clang++-14), nvcc 12.8

Les calculs s'effectueront sur des maillages de plus en plus raffinés, jusqu'à ce que le temps d'execution dépasse 10 secondes ou que le résultat soit erroné. Le classement final sera basé sur une estimation de la taille du problème que le programme peut résoudre en 10 secondes.[2]

Le format de soumission sera sous la forme d'un zip contenant un fichier `CMakeLists.txt` et les sources nécessaires pour compiler votre programme sous forme d'une librairie partagée. Un exemple ainsi que

[2] L'estimation sera basée sur une interpolation log-log entre les deux derniers points (le dernier < 10s et le premier > 10s).

toutes les éventuelles modifications de règles seront disponible sur git. En cas de doute ou questions concernant n'importe quelle partie du concours, n'hésitez pas à contacter les assistants Nathan Tihon ou Miguel De Le Court.

Il n'est pas indispensable d'avoir une version GPU et parallèle pour participer au concours : un bon code séquentiel et efficace sera déjà apprécié. La participation au concours est facultative et les points acquis sur cette partie seront du pur bonus. Il est théoriquement possible d'obtenir 20/20 à l'examen sans participer au concours, même si cela risque d'améliorer légèrement votre code.

Pour le concours, on fournira un exemple précis de maillage et de problème à résoudre : on se restreindra à des éléments triangulaires linéaires et à de l'élasticité linéaire en tensions ou déformations planes. Il est donc possible d'avoir une version optimisée sans les options supplémentaires que vous auriez implémenté dans le projet.

2.8 Echéances

Définition du problème	
Soumission du texte d'une page	Vendredi 21 février
Approbation finale par l'assistant de référence	Vendredi 28 février
Soumission du projet	
Soumission du code du projet	Vendredi 28 mars - Vendredi 4 avril
Interview sur le projet	
Interview avec votre assistant de référence...	Lundi 7 avril - Lundi 14 avril
et démonstration de votre programme	Mardi 8 avril - Mardi 15 avril
Soumission d'un code pour le grand prix de l'élément le plus fini	
Soumission du code optimisé	Lundi 5 mai

2.9 Mon binôme a disparu....

Si pour une raison précise, vous ne souhaitez pas faire le projet avec votre binôme. Il faut envoyer une-mail au titulaire du cours ainsi qu'à votre compagnon de travaux. Il faut aussi veiller à envoyer un email à votre binôme défaillant pour l'informer de votre *mise en demeure* de participer au projet. La note du second étudiant sera alors nulle pour le projet. Si cet étudiant s'est désinscrit du cours ou a clairement abandonné son année d'étude, cela ne devrait poser aucun problème pour lui !

Attention : c'est le seul cas de figure admis. Si votre binôme souhaite réellement participer au projet, vous ne pouvez pas l'exclure de manière unilatérale. Mais si sa participation est réellement inexistante ou nulle, vous le mentionnez dans le rapport du projet et nous en tiendrons compte dans la note du projet. Deux étudiants isolés peuvent remettre un projet commun : il suffit aussi de le mentionner dans le rapport soumis. Vous indiquerez clairement dans votre rapport que vous avez fait le projet seul.