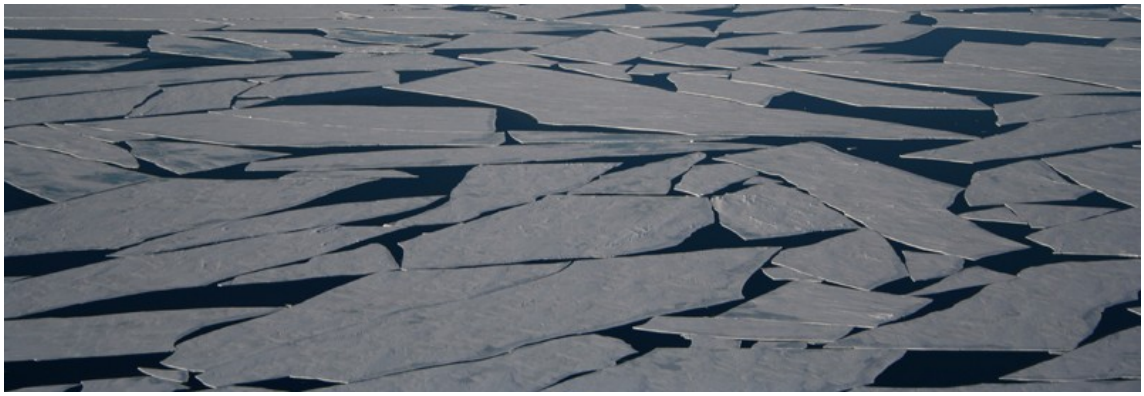
Rapport de stage

Fracturation de floes de glace par percussion dans un modèle granulaire

Étudiant
Roussel Desmond Nzoyem

Superviseur
Stéphane Labbé

Enseignant référent
Christophe Prud'homme



Stage effectué au Laboratoire Jacques-Louis Lions ;
du 03 février 2021, au 31 juillet 2021 ;
pour l'obtention du master 2 CSMI.

Année académique 2020 - 2021

14/8/2021

Remerciements

Avant tout développement sur cette expérience professionnelle, il apparaît opportun de commencer ce rapport de stage par des remerciements, à ceux qui mont appris des choses, et à ceux qui ont eu la gentillesse de faire de ce stage un moment agréable et profitable.

Ainsi, je remercie le Pr. Stéphane Labbé, mon maître de stage qui ma formé et accompagné tout au long de cette expérience avec beaucoup de patience et de pédagogie. Étant donné la situation sanitaire de COVID-19, il a su me transmettre tous les enseignements et les ressources (livres, reférences, etc.) nécessaires pour effectuer mes différentes missions (et bien plus encore), à distance comme en présentiel. Je vous en suis profondément reconnaissant.

J'éttends mes remerciements à mes illustre prédécesseurs Matthias Rabatel et Dimitri Balasoiui sans qui mon travail n'aurait pas eu lieu. Dimitri a su me guider dans les moments les plus difficile du stage. Les mots ne sauraient exprimer ma reconnaissance envers les visio-conférence organisées afin de me permettre de prendre en main de son travail.

Je remercie aussi l'ensemble du personnel du Laboratoire Jacques-Louis Lions qui ma permis deffectuer un stage scientifique très enrichissant dans les meilleures conditions possibles. J'addresses mes salutations aux doctorants et aux étudiants en séjour de recherche pour leur unique regard sur les difficultés auquelles fait face. En particulier, je remercie Madame Catherine Drouet de l'administration pour son assistance et ses conseils inestimables.

Enfin, je remercie mes proches, ma famille et mes amis pour leurs encouragements. Si un lecteur estime que son nom aurait du figurer ici de facon explicite, faite ceci : imprimer cette page, montrer la moi, et cela sera votre coupon pour une bièrre gratuite (ou un caffé, ou autre chose). Les trucs gratuits sont mieux qu'une mention, n'est-ce pas ?

Table des matières

Chapitre 1

Problème 1D et étude de la fracture

1.0.1 Une approache combinatoire pour la fracture

Etant donné le déplacement des noeuds du floes à un temps précis t^{n+1}, nous recherchons la fracture qui minimise l'énergie potentielle totalle au temps t^{n+1} :

$$\boxed{\begin{aligned} E^{n+1} \quad &= \text{énergie potentielle élastique au temps } t^{n+1} \\ &+ \text{ ténacité} \times \text{longeur de la fracture envisagée au temps } t^{n+1}. \end{aligned}} \tag{1.1}$$

La longeur de la fracture correspondant ici à la longueur à vide du ressort[1] à cette position si celui ci s'était brisé. Notons que nous pouvons remplacer l'énergie potentielle élastique dans l'équation (1.1) par la l'énergie de déformation du matériau (énergie potentiele élastique + énergie dissipée par frotemment visqueux). Notre approche combinatoire consite ainsi à calculer les énergie correspond à toutes les fractures possibles dans le floe, et nous récupérons la plus petite de ces énergies E^{n+1}_{min}. Conformément à l'approche de Griffith, une fracture n'apparait que si $E^{n+1}_{min} \leq E^{n}_{min}$.

L'approche combinatoire n'est supportable qu'en 1D car à chaque pas de temps, il y a autant de fracture admissibles qu'il ya de ressorts[2]. En effet, à chaque ressort brisé, c'est un nouveau floe qui est créé. Cette simplication élimine aussi le problème de ***propagation de la fracture***; en 1D, nous ne traiterons donc que le problème de ***nucléation de la fracture***.

Tous ceci n'est évidement pas le cas en dimension supérieur, et nous devons étudier toutes les combinaisons psossibles de ressorts qui cèdent simultanément; ce qui est très couteux. Il est donc primordial d'implémenter la méthode du champ de phase pour toute adaptation de nos résultats en dimension 2 ou 3.

1.1 Algorithme de calcul 1D

Nous présentons dans cette section l'algorithme que nous avons adopté pour développer notre code de simulation de la percussion ainsi que la fracture de floes 1D. Notons que l'implémtation de la percussion correspond au modèle présenté à la **??**. Cependant, les autres modèles de percussion peuvent facilement etre obtenus en changeant le contenu de la fonction `computeAtContact` que nous détaillerons plus tard. Quant à la fracture, nous avons implémenter l'approche combinatoire présentée à la section 1.0.1. Un résumé de la procédure dans l'intervalle de temps $[t^n, t^N]$ est présenté à la **??** 1.

Notons que l'algorithme **??** 1 omet plusieurs détails indispensables pour une implémentation efficace en language Python. Par exemple, l'invalidation des quantitées est prise en main par les fonctions de détection

1. Nous utiliserons la désignation ***ressort*** ici pour indiquer à la fois le ressort et le dispositif visqueux situé entre deux neouds adjacents (VOIR FIGURE SUR LA PARTIE PRÉCÉDENTE)

2. En réalité, les deux ressorts exptrèmes ne peuvent se briser, car cela donnerait naissance à des floes sans ressorts

Algorithme 1 : Algorithme de simulation de percussion et de fracture

Données : Les positions $\mathbf{q}^n$, les vitesses $\mathbf{v}^n$ de tous les noeuds de tous les floes à l'instant t^n sont connus [3] ; Mettre à jours les chargements aux bords au temps courant t^{n+1} ; Pour chaque neud e, début des vérifications de collsion à partir de $Coll_e = 0$; Pour chaque floe f, une fracture potentielle apparait au temps $Frac_f = N$;

Résultat : Positions et vitesses des noeuds à l'instants t^N

If$n = 0$ (premier pas de temps) Calcul des trajectoires des noeuds $\mathbf{q}^{n+1}, \dots \mathbf{q}^N$ et des vitesses associées $\mathbf{v}^{n+1}, \dots \mathbf{v}^N$; Vérification de l'apparition d'au moins une collision ;

tant que ***(pas de collision)*** **faire**

doubler le temps de simulation ; recalculer le trajectoires ; pour tous les noeud e, $Coll_e = k$ (k est le pas de temps courant) ; revérifier la collision ;

fin

tant que ***(*$Coll_e < N$ *pour tous les noeuds* e*)*** **faire**

Calcul des déplacements après chocs ; Détection des fractures : pour chaque pas de temps $t^k \in [t^{n+1}, t^N]$ et pour chaque floe, rechercher la configuration qui présente l'énergie potentielle totale minimale E^k_{min}, et la comparer avec l'énergie E^n_{min} ;

si ***(fracture sur le floe*** f***, i.e.*** $E^k_{min} \leq E^n_{min}$**) alors**

$Frac_f = k$;

sinon

$Frac_f = N$;

fin

Détection des collisions : pour chaque pas de temps $t^k \in [t^{n+1}, t^N]$ et pour chaque noeud, on se demande s'il y a collision avec le noeud voisin de droite ;

si ***(collision entre*** e_1 ***et*** e_2***)*** **alors**

$Coll_{e_1} = Coll_{e_2} = k$;

sinon

$Coll_{e_1} = Coll_{e_2} = N$;

fin

Mise à jour des indices potentiels de fracture : tous ces indices sont reinitialisés au minimum de tous les indices de fracture $Frac = \min_{f \in \mathcal{F}} Frac_f$, où $\mathcal{F}$ désigne l'ensemble des floes ;

Mise à jour des indices potentielles de collision : Désignons par $Coll'$ le minimum de tous les indices de collisions $\min_{e \in \mathcal{E}} Coll_e$, où $\mathcal{E}$ désigne l'ensemble des neouds du problème. Tous les indices sont reinitialisés à $Coll = \min\{Coll', Frac\}$.

Invalidation des quantités : toutes les positions et les vitesses calculées qui survienne après une collision ou une fracture sont invalidées (cependant, les nouveaux calculs ne débutent qu'à partir de $Coll$, et s'arètent à $Frac$).

fin

des collision et de fracture. Nous fournissons donc la méthode `runSimulation()` de la classe FractureSolver (voir liste 1.1).

```
    def runSimulation(self):
        """
        Runs the simulation for the complete fracture problem
        """

        ## Run uniform mouvement phase up to the first collision
        self.computeBeforeContact()

        for key in self.floes.keys():
            self.checkFracFrom[key] = self.t.size

        while self.t.size <= self.NBef + self.NAft and max(self.collCount.values()) < 1000:

            self.computeAfterContact()

            ## (Potential) Fracture detection
            floeDict = deepcopy(self.floes)
            for floe in floeDict.values():
                self.checkFracture(floe.id)

            ## Collision detection
            for floe in self.floes.values():
                for node in floe.nodes:
                    res = self.checkCollision(node.id, node.rightNode)

            ## Assign the same value to all keys to check collision from now on
            smallestCheckCollFrom = min(self.checkCollFrom.values())
            for key in self.checkCollFrom.keys():
                self.checkCollFrom[key] = smallestCheckCollFrom
            self.collLoc.append(smallestCheckCollFrom)

            ## Assign the same value to all keys tp check fracture from now on
            smallestCheckFracFrom = min(self.checkFracFrom.values())
            for key in self.checkFracFrom.keys():
                self.checkFracFrom[key] = min([smallestCheckFracFrom, smallestCheckCollFrom])

        ## Truncate exeeding data (not strickly necessary)
        self.x = self.x[:self.NBef+self.NAft+2, :]
        self.v = self.v[:self.NBef+self.NAft+2, :]
        self.t = self.t[:self.NBef+self.NAft+2]
```

Listing 1.1 – Code de simulation 1D.

Notre code est stocké dans un dépot GitHub privé dont une explication du contenu est donnée dans le fichier README dont la capture est présentée à la FIGURE. Nous avons effectué plusieurs simulations avec ce code. En particulier, nous avons observé que que les ressorts des floes se fractures quand il sont soumis soit à une compression ouà une élongation intense. Nous pouvons oberver celà sur cette simulation LIEN SEAFILE, fournie à titre d'exemple.

README DU REPO GITHUB

1.2 Résumé des résultats obtenus

Pour le problème 1D, nous avons non seulemetn étudié le déplacement des noeuds d'un floe, mais aussi ce qui se passe après une collision de ces noeuds. En considérant le floe comme un réseau de ressorts, nous avons pu étudier la nucléation (et la propagation) d'une fracture suivant le modèle de Griffith sans recourir à la méthode du champ de phase.

Nous avons éffectué plusieurs simulation qui ont montrer que les ressorts se brisent lorsqu'ils sont fortement compréssés. Le modèle 1D adopté (voir **??**) a également montré que le système perds de son énergie (cinétique) au cours du temps, ce qui

Quand à la validation des résultats, nous n'avons pas réussi à effectuer des tests en laboratoire. Cette tache représente la prochaine étape avant l'adoption de ce modèle 1D. Ceci dit, les floes de glace sont généralement considérés comme des objets 2D du fait de leur taille négligeable face au rayon de la terre. Une étude en dimension supérieure est donc indispendable pour un déployment de notre modèle de fracture à l'échelle des floes de glace.