Mini-projet de programmation

Modèles d'avalanche

Au cours de cette séance, nous allons étudier un exemple de modèle dynamique mettant en jeu un système critique auto-organisé. Ce type de modèle et cette notion ont été introduits en 1987 par Bak, Tang et Wiesenfeld[1]. Une illustration de ce genre de modèles est la dynamique du "tas de sable". Si on ajoute des grains très lentement sur un tas de sable, on observe le déclenchement d'avalanches : dès que la pente locale du tas est supérieur à une valeur critique, l'empilement est instable, les grains se réorganisent et engendre un mouvement collectif, c'est à dire une avalanche. Ce type de dynamique est très délicat à appréhender puisqu'il faudrait décrire le mouvement de chacun des grains, lesquels sont régis par des forces de contact solide. Une autre approche est donc nécessaire, et c'est un des objectifs du modèle que nous allons étudier. Même si ce modèle ne décrit que mal la réalité des écoulements granulaires car il ne prend pas en compte l'inertie ni la dissipation du mouvement des grains, il apporte des concepts intéressants comme la notion d'état critique auto-organisé et a suscité un très grand engouement dans la communauté scientifique. Il peut en effet aussi s'appliquer à d'autres phénomènes implicant une propagation d'un seuil critique, comme les tremblements de terre ou bien encore les feux de forêt.

Pour rendre l'algorithme plus concret, envisageons le modèle à une dimension. On considère N sites de hauteur h_n. Chacun de ces sites contient h_n grains de sables. La pente locale z_n du tas peut se calculer comme $z_n = h_n - h_{n+1}$. Si on ajoute un grain au hasard sur le site n, la pente locale des sites n et $n-1$ est modifiée. Aussi, on met à jour les valeurs de h_n et de z_n de la manière suivante :

$$\left| \begin{array}{rcl} h_n & \rightarrow & h_n + 1 \\ z_n & \rightarrow & z_n + 1 \\ z_{n-1} & \rightarrow & z_{n-1} - 1 \end{array} \right. \tag{1}$$

On teste ensuite les valeurs de z_n. Si z_n est supérieur à une pente critique z_c (par exemple 2), alors le grain ajouté est déplacé sur le site voisin (on suppose pour simplifier que les grains ne bougent que de gauche à droite), ce qui engendre des modifications des pentes sur les sites n, $n+1$ et $n-1$. Plus précisément, cette redistribution conduit aux mises à jour suivantes :

$$\left| \begin{array}{rcl} h_n & \rightarrow & h_n - 1 \\ h_{n+1} & \rightarrow & h_{n+1} + 1 \\ z_n & \rightarrow & z_n - 2 \\ z_{n\pm1} & \rightarrow & z_{n\pm1} + 1 \end{array} \right. \tag{2}$$

Cette réorganisation peut provoquer un nouveau dépassement de la pente critique en z_{n+1} (ou en z_{n-1}). Dans ce cas, on réitère l'épate précédente, le grain passant sur le site $n+2$. Il peut à nouveau y avoir un dépassement de la pente critique, etc ... S'il n'y a plus de dépassement de pente, alors on revient à l'étape 1, qui consiste à simplement ajouter un grain sur un site choisi au hasard. Si un grain arrive en N et dépasse le seuil, alors il est définitivement perdu.

Les résultats de cet algorithme sont triviaux (et il n'y a pas besoin de faire tourner l'algorithme pour le montrer !). Il existe en effet un état critique : celui qui satisfait $z_n = z_c$, pour toutes les valeurs de n. Dans cet état, la moindre perturbation du système (l'ajout d'un nouveau grain) conduira automatiquement à une avalanche jusqu'en $z = N$ où le grain sort de la boîte [2].

A deux dimensions en revanche, ce type d'algorithme conduit à des résultats non triviaux. L'objectif de ce mini-projet est de programmer cet algorithme a deux et à trois dimensions, et d'en

[1] Ce sujet est inspiré de l'article suivant : Bak, P., Tang, C, Wiesenfeld, K, *Phys. Rev. A*, **38**, 364 (1988)

[2] Il existe une autre manière d'utiliser l'algorithme : on choisit au départ une forme au hasard du tas (donc une série h_n) dont tous les z_n sont au dessus du seuil critique. Dans ce cas, on va avoir de nombreuses avalanches en série, qui ne vont s'arrêter que lorsqu'on aura atteint l'état critique où tous les z_n sont égaux à z_c.

extraire les résultats.

Nous utiliserons Matlab pour programmer. Vous sauvegarderez au fur et à mesure les scripts utilisés et les figures qui vous sont demandées, ainsi qu'un document texte où vous répondrez aux questions qui vous sont posées. L'objectif est d'aller le plus loin possible en répondant avec soin aux maximums de questions. Il sera tenu compte dans la notation de la clarté du code (pensez à les commenter !) et de son efficacité, mais également de la qualité des interprétations et réponses fournies ainsi que de la clarté des figures produites.

1 Avalanches à deux dimensions.

A deux dimensions, le traitement de la pente critique est plus délicat qu'à une dimension, puisqu'il faudrait tenir compte de la direction de la pente. Nous allons simplifier le problème en ne tenant pas compte de la direction, ce qui nous éloignant un peu de l'application concrète du tas de sable, mais qui permet toutefois d'obtenir des résultats intéressants.

Lorsqu'on la pente $z_{i,j}$ dépasse en un point i, j du réseau une valeur critique z_c, on applique la règle de redistribution suivante :

$$\left| \begin{array}{rcl} z_{i,j} & \rightarrow & z_{i,j} - 4 \\ z_{i\pm1,j} & \rightarrow & z_{i\pm1,j} + 1 \\ z_{i,j\pm1} & \rightarrow & z_{i,j\pm1} + 1 \end{array} \right. \tag{3}$$

Il s'agit donc d'une redistribution locale de la pente. Le lien avec une hauteur h du tas de sable n'est pas triviale, et nous chercherons plus à le faire dans ce cas.

Comme précédemment, cette redistribution peut entraîner un dépassement du seuil critique sur les voisins du site i, j, auquel cas on applique à nouveau la règle de redistribution de l'équation 3 sur les voisins, et ainsi de suite.

Lorsqu'il n'y a plus de redistribution, on choisit un site au hasard i, j et on incrémente la pente locale :

$$z_{i,j} \rightarrow z_{i,j} + 1 \tag{4}$$

1.1 Coeur de l'algorithme

#1 Créer un fichier pour le script principal. Dans ce script, commencer par créer une constante N qui représente la taille du réseau, initialisée à 50, une variable t qui représente le temps de la simulation, une constante z_c qui respésente le seuil critique et construire une matrice z de dimension $N \times N$ remplie par des zéros uniquement.
Indication : On rappelle l'existence de la fonction `zeros` qui permet de créer des vecteurs à plusieurs dimensions. La variables t et le réseau z pourront être déclarées comme variables globales, afin qu'elles puissent être utilisées par toutes les fonctions crées dans la suite.

#2 Créer une fonction `increment` permettant de réaliser l'équation 4 sur un site choisi au hasard. Cette fonction devra aussi effectuer un test pour savoir si l'on dépasse localement le seuil critique z_c, défini à 4. Si $z_{i,j} \geq z_c$, on appelle la fonction `redistribution`, qui sera créée dans la question suivante. Cette fonction doit incrémenter le temps t de la simulation.
Indication : La fonction `rand` permet de générer des nombres aléatoires.

#3 Créer une fonction `redistribution` permettant de réaliser l'équation 3 sur un site i, j ayant dépassé le seuil critique. De même que dans la fonction `increment`, il est nécessaire à chaque fois qu'on met à jour une valeur $z_{i,j}$ de faire un test pour savoir si la valeur $z_{i,j}$ dépasse le seuil fixé z_c. Auquel cas, il faut appliquer à nouveau la redistribution. Les conditions aux limites doivent être fixées dans cette fonction : on utilisera des conditions aux limites de sortie libre (la redistribution ne s'effectue que sur les voisins qui sont dans la boîte de sinmulation). Par exemple, si $j = N$ et $1 < i < N$ on obtient :

$$\left| \begin{array}{rcl} z_{i,N} & \rightarrow & z_{i,j} - 4 \\ z_{i\pm1,N} & \rightarrow & z_{i,j} + 1 \\ z_{i,N-1} & \rightarrow & z_{i,N-1} + 1 \end{array} \right. \tag{5}$$

On perd dans ce cas une "particule". Si on se trouve dans un coin (par exemple $i = N$, $j = N$, on perd 2 "particules".

Indication : La fonction `redistribution` est une fonction récursive.

A ce stade, nous avons le minimum pour faire tourner l'algorithme. Il suffit d'appeler autant de fois que souhaité la fonction `increment` pour réaliser l'expérience. Une méthode est d'utiliser une boucle conditionnelle sur le temps de la simulation.

#4 Créer une constante t_{max} en début de script, et ajouter une boucle conditionnelle ($t < t_{max}$) à l'intérieur de laquelle se trouve un appel à la fonction `increment`.

1.2 Observables

Il est nécessaire pour pouvoir aller plus loin de définir et d'incorporer des observables. Il y a plusieurs grandeurs qu'il est intéressant de mesurer : le nombre de "particules" s qui sortent de la boite entre deux incrémentations et le nombre de redistributions r entre deux incrémentations. On remarquera que le temps t compte le nombre de particules ajoutées.

#5 Incroporer dans le code la mesure de $s(t)$ et de $r(t)$.

Indication : Il est préférable pour ces variables de les déclarer comme variables globales afin de pouvoir y accéder partout, et en particulier après l'exécution du script afin de pouvoir les utiliser.

#6 Vérifier que le nombre de particule est conservé. Plus concrètement, l'égalité suivante doit être vérifiée à tout instant :

$$\sum_{i,j} z_{i,j}(t) = t - \sum_{t'=0}^{t} s(t') \tag{6}$$

1.3 Etat Stationnaire et taux de remplissage critique

#7 Calculer une nouvelle observable $Z(t) = \sum_{i,j} z_{i,j}$. L'intérêt de cette variable est de montrer que l'on atteint un état stationnaire au bout d'un certain t : en moyenne, il y a autant de particules qui rentrent que de particules qui sortent. Faîtes tourner l'algorithme sur un intervalle de temps de 8000, et tracer $Z(t)$. Sauvegardez cette figure. Donner une estimation du temps t_0 nécessaire pour atteindre l'état stationnaire. Calculer avec une bonne précision la valeur moyenne $\langle Z \rangle$ de $Z(t)$ dans l'état stationnaire (calculer la moyenne sur un intervalle de temps de 2000 au minium).

Indication : Assurez-vous que votre algorithme ne comporte pas d'erreurs avant de lancer les 8000 itérations qui peuvent être un peu longues...

On appelle "taux de remplissage critique" le rapport $\langle Z \rangle / Z_{max}$, Z_{max} étant le remplissage maximum du réseau. Que vaut Z_{max} ?

#8 Déterminer le taux de remplissage critique pour $N = 100$, $N = 50$, $N = 30$, $N = 20$, $N = 10$. Tracer ce taux en fonction de N afin de mettre en évidence qu'il tend vers une valeur fixe. Justifier que ce taux augmente avec la taille du réseau.

Indication : Estimez avant de faire tourner l'algorithme le temps t_{max} nécessaire dans chacun des cas. On utilisera pour cette estimation le fait que le taux de remplissage critique ne dépend que très peu de N.

Dans tout ce qui suit, nous allons nous intéresser uniquement au systèmes obtenus à $t > t_0$. Puisque l'on a atteint un état stationnaire, il n'est plus utile de réinitialiser à chaque fois le réseau $z_{i,j}$, ce qui permet de gagner en temps de calcul.

#9 Il est intéressant d'observer qualitativement la forme des avalanches crées. Pour observer une avalanche, il faut appliquer une perturbation (ie : appliquer l'équation 3), et sauvegarder les sites i, j qui ont subi des modifications à cause de la perturbation initiale. On distinguera les sites sur lesquels il y a eu dépassement du seuil critique des sites qui ont été modifiés par l'avalanche sans dépassement du seuil. Afficher ensuite une image de ces sites (sites avec dépassement de seuil en blanc, sites avec une modification en gris, et sites non affectés en noir). Enregistrer 4 figures de ce type : `cluster1`, `cluster2`, ... si possible illustrant 4 tailles très différentes.
Indication : Utilisez la fonction `imagesc`.

1.4 Distributions de s et de r

#10 Tracer deux figures, *dans l'état stationnaire*, illustrant s et r en fonction de t sur un intervalle de t de 2000. Sauvegarder ces figures.

#11 Dans l'état stationnaire, calculer puis tracer la densité de probabilité de r, notée $P(r)$. Faire attention à ce que ce calcul soit réalisé sur un nombre de données suffisantes pour que les moyennes soient représentatives. La fonction $P(r)$ sera tracée en log-log.

Indication : La densité de probabilité est obtenue en normalisant à 1 un histogramme des valeurs de r. La difficulté de la question réside dans le bon choix des intervalles pour l'histogramme. En effet, si les intervalles sont trop petits, il n'y aura pas assez de données pour calculer un histogramme représentatif. S'ils sont trop grands, la précision sera moins bonne. Une bonne méthode consiste à travailler avec des intervalles variables, de telles sorte qu'il y ait environ le même nombre de point dans chacun des intervalles. Puisqu'il y a beaucoup d'avalanche de petite taille et peu d'avalanche de grande taille, les intervalles doivent donc être de taille croissante. On pourra choisir une taille d'intervalle qui varie suivant une loi de puissance : $l_i \sim i^{1.5}$, où l_i est la longueur de l'intervalle i. Attention, il faut évidemment renormaliser les résultats de l'histogramme pour tenir compte du fait que les intervalles ne sont pas de taille constante. Les histogrammes devront être obtenus sur environ 10^4 points.

#12 Montrer que $P(r)$ suit une loi de puissance : $P(r) \sim r^{\alpha}$, sauf pour les plus grandes valeurs de r. Déterminer l'exposant α en ajustant les données de la simulation (on prendra soin d'exclure les grandes valeurs de r) et ajouter le tracé de l'ajustement sur la figure obtenue à la question précédente. Pourquoi ces lois ne sont-elles pas vérifiées pour les grandes valeurs de r ?

Indication : Il n'est pas utile d'utiliser les fonctions d'ajustement non linéaire ... Par ailleurs, il convient d'exclure des histogrammes les cas où $r = 0$ qui ne sont pas pertinents.

1.5 Dynamique

Jusqu'à présent, nous nous sommes intéressés à la taille des avalanches. Dans ce qui suit, nous nous intéressons à leur dynamique. Il est pour cela nécessaire de redéfinir l'échelle de temps. En effet, avec l'échelle de temps utilisée jusqu'à présent, une avalanche est instantanée.

L'échelle de temps que nous introduisons ici représente le nombre d'enchaînements successifs des distributions selon un chemin. Supposons qu'on incrémente le site i, j et que celui-soit instable. La redistribution affecte d'abord les premiers voisins. Cette étape est notre nouveau pas de temps élémentaire. Sur les premiers voisins, des redistributions peuvent avoir lieu en même temps, et un nouveau pas de temps conduit l'avalanche à progresser jusqu'aux deuxièmes voisins. Supposons que des redistributions ont lieu jusqu'au n-ième voisin, la durée de l'avalanche sera ainsi n. La nouvelle échelle de temps, notée t_p correspond donc au nombre de récursions effectuées. Ainsi définie, la durée T de l'avalanche est au maximum égal au nombre de sites impliqués dans l'avalanche, mais sera généralement beaucoup plus petite.

Avec cette nouvelle échelle de temps, on peut calculer la durée T d'une avalanche, mais on peut également aller un peu plus loin en considérant que le nombre de redistributions correspond à l'énergie dissipée pendant l'avalanche (pour le tas de sable, cette dissipation provient des frottements engendrés par le mouvement des grains). Ainsi le taux de croissance de l'avalanche $r(t_p)$ représente l'énergie dissipée pendant l'avalanche. C'est une grandeur dynamique, et on s'intéresse à ses propriétés.

#13 | Implémenter dans l'algorithme la mesure de $r(t_p)$ et de T, pour chaque avalanche. Tracer quelques exemples de $r(t_p)$.

1.5.1 Spectre de fréquence

Pour caractériser la dynamique du système, on considère qu'il est soumis à des perturbations aléatoires dans l'espace et dans le temps et on s'intéresse à sa réponse, c'est à dire ici à l'énergie dissipée totale. Les perturbations étant aléatoirement réparties dans le temps, l'espace des temps n'est pas très pertinent. En revanche, dans l'espace des fréquences, l'origine des temps n'importe pas et on s'attend à ce que le spectre de fréquence associé à se signal $r(t_p)$ pseudo-aléatoire soit caractéristique du système considéré. On peut donc moyenner les spectres des fonctions $r(t_p)$ pour caractériser la dynamique de notre modèle.

#14 | Afin de caractériser ces fluctuations des fonctions $r(t_p)$, déterminer le spectre de puissance $R(f)$, où f est la fréquence, de $r(t_p)$. Moyenner les spectres sur environ 1000 avalanches, et tracer la moyenne obtenue en log-log. Pourquoi observe-t'on un palier pour les fréquences les plus faibles ?

Indication : Le spectre de puissance R est définit par : $R = TF(r) \times TF^*(r)$, où TF est la transformation de Fourrier (* étant le conjugué). Utiliser la fonction `fft` qui réalise l'algorithme de transformée de Fourrier rapide. Cet algorithme est efficace sur des valeurs discrètes de taille $n = 2^p$. Il est donc recommandé de travailler avec des vecteurs de longueur dont la taille est une puissances de 2. Attention au résultat obtenu : la transformée de Fourrier de fonctions réelles étant symétrique, seule les $n/2$ premières valeurs sont intéressantes. On fera également attention aux abcisses : si les données temporelles ont un pas de temps de Δt, le pas en fréquence de la transformée de Fourrier sera $1/(n\Delta t)$.

#15 | Sauf pour les fréquences les plus faibles, le spectre $R(f)$ est décrit par une loi de puissance : $R(f) \sim f^{-\delta}$. Determiner l'exposant δ.
Quel serait le spectre de puissance d'un bruit blanc ? Quel serait le spectre de puissance d'un systême possédant un seul temps de relaxation ? Qu'en déduisez-vous quant à la dynamique du système considéré ?

1.5.2 Durées des avalanches

#16 Déterminer la loi distribution $p(T)$ de T. Malgré la faible amplitude de l'intervalle de valeurs de T, on peu supposer que $p(T)$ suit une loi de puissance : $p(T) \sim T^{\beta}$ (il faudra pour mieux mettre en évidence augmenter la taille N du réseau). Estimer l'exposant β.

#17 Montrer que T et r sont corrélés, et que $T \sim r^{\gamma}$. Déterminer l'exposant γ. Cette corrélation permet de définir une relation théorique entre les exposants γ, α et β :

$$p(T) = p(r)\frac{dr}{dT}$$

Dériver la relation attendue entre γ, α et β et vérifier que cette relation est vérifiée sur les valeurs issues de la simulation. Expliquer les différences éventuelles.

2 Avalanches à trois dimensions.

#18 Implémenter l'algorithme à trois dimensions en utilisant un réseau de $20 \times 20 \times 20$ et comparer les résultats obtenus (taux de remplissage critique, distribution de s, corrélation entre s et T) avec ceux obtenus à deux dimensions.
Indication : La réorganisation se fait maintenant sur 6 voisins ! Il convient donc de modifier le seuil critique et de réécrire l'équation 3