Rapport de projet expérimental

Département de Physique de l'ENS

Cristaux de Wigner

Elèves :
Elie Gouzien
Sophie Marbach

Supervision :
B. Peaudecerf

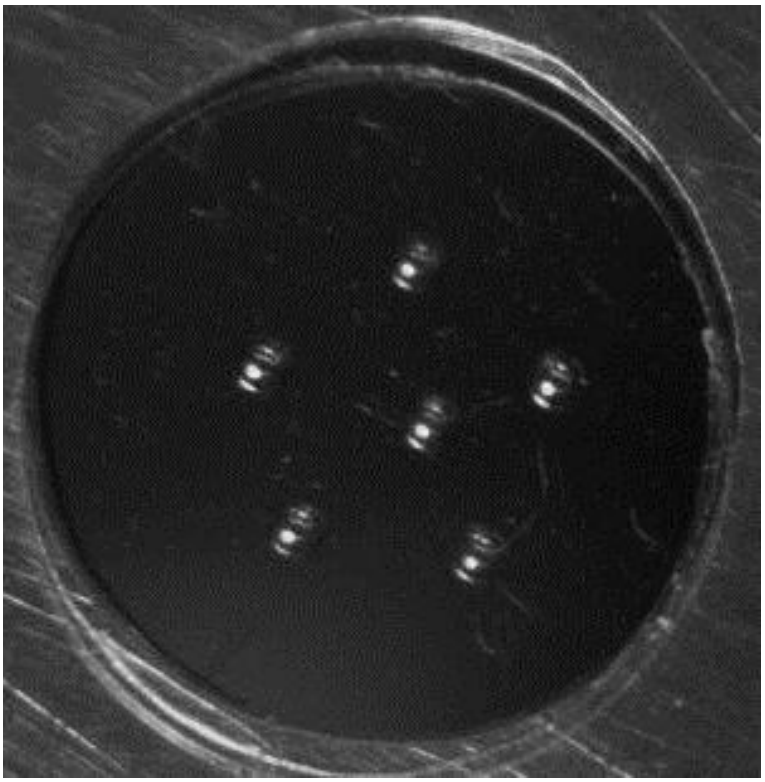

Figure 1 – Interaction de six billes caractérisant un cristal bidimensionnel à six billes dans sa configuration le plus stable, en forme de pentagone

Février-Juin 2012

Table des matières

Introduction

On appelle Cristaux de Wigner des cristaux pouvant être décrits par un mouvement caractéristique à deux dimensions d'espace. Les représentations de cristaux bidimensionnels se trouvent dans de nombreux autres systèmes physiques tels que les réseaux de vortex ou les cristaux colloïdaux. Tout l'enjeu est de comprendre la réaction de cristal face à des contraintes extérieures.

L'étude physique de tels cristaux peut se faire par des "modèles réduits", dans notre cas il faudrait dire des modèles agrandis. L'objet de notre projet expérimental était l'étude d'une modélisation cristalline bidimensionnelle inspirée de la thèse de Gwennou Coupier [1] à ce sujet.

Pour modéliser un cristal 2D de façon expérimentale, il faut modéliser des atomes (on peut commencer par des atomes tous identiques) en interaction répulsive (répulsion des couches supérieures d'ions). Il faut également modéliser l'agitation thermique des atomes autour de leur position d'équilibre (tout cristal étudié dans un milieu naturel est soumis à l'agitation thermique du milieu qui l'entoure (ne serait-ce que des molécules d'air). Enfin il faut s'assurer que le tout a lieu dans un plan et que le système n'est en rien affecté par la surface sur laquelle il repose.

Description du système expérimental On nous avait fourni le modèle suivant : sur une planche principale, on vissait une plaque secondaire. Au milieu de cette plaque était un support astucieux constitué d'une première couche de Silicium, une couche de vide dans lesquels étaient délicatement mises de minuscules billes (les atomes) en métal, et au dessus une couche conductrice transparente pour les enfermer et refermer le condensateur ainsi crée. La planche conductrice pouvait être alimentée par une source de très haute tension (1000 V) ce qui chargeait les bords de chaque trou et les billes négativement, pour tenter de recréer l'interaction entre atomes. Deux hauts-parleurs fixés sur deux coins de la planche principale faisaient vibrer le tout, créant un bruit "blanc" (fréquences d'égales puissance entre 20 et 200Hz) commandés en tension par l'ordinateur.

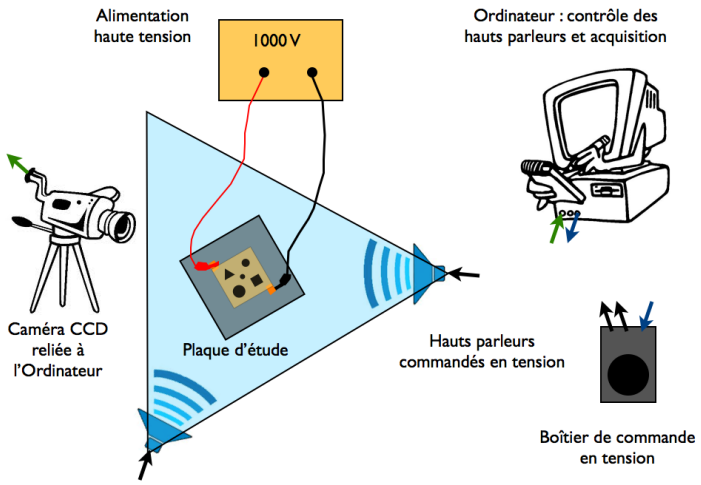

FIGURE 2 – Schéma général du dispositif

Objectifs

- Caractériser la notion de "température" dans ce modèle et déterminer une loi qui caractériserait l'évolution de la "température".
- Étudier le comportement de six billes dans un disque, et associer des lois d'évolution pour ces différents comportements (durée de vie d'un état, différence d'énergie entre deux états, énergie de l'état de transition).
- Étudier le comportement de deux billes dans un puits de potentiel afin de trouver le potentiel associé à la répulsion entre deux billes.

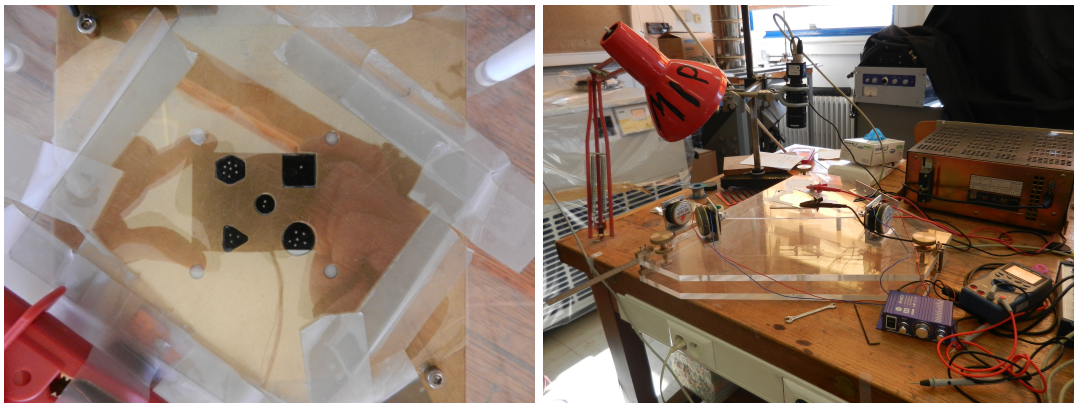

FIGURE 3 – Photographies du dispositif expérimental

I Création d'un bain thermique

Quand l'on cherche à étudier un cristal, on étudie de petits atomes qui oscillent autour de leur position d'équilibre pour cause de différents facteurs, et primordialement, en l'absence d'environnement extérieur particulier (pas d'application de champ électrique par exemple), le cristal est juste dans l'air. L'air se trouve à une certaine température, c'est-à-dire que les molécules constituant l'air se meuvent à une certaine vitesse quadratique moyenne, et percutent les atomes en surface du cristal, qui eux-même rentrent en collision avec les atomes plus à l'intérieur. Somme toute, chacun des atomes du cristal est ainsi "soumis" à ce qu'on appelle "l'agitation thermique".

Le montage que nous avons effectué, pour modéliser proprement un cristal, doit pouvoir reproduire ce mouvement aléatoire, et reproduire les mêmes variations que des molécules soumises à l'agitation thermique lorsqu'on fait varier la température, soit pour notre montage, lorsqu'on fait varier l'agitation (le bruit).

I.1 Présentation du dispositif recréant le bain thermique

Les billes étant cantonnées à un plan horizontal. On peut considérer que les billes ne peuvent bouger que selon deux degrés de libertés qui seront appelés x et y dans la suite.

A deux coins adjacents de la planche on place des hauts parleurs liés à la planche et orientés de façon orthogonale. Ceux ci sont reliés à un amplificateur puis à une interface numérique, qui nous permet de régler les caractéristiques de l'excitation depuis l'ordinateur.

Le principe est d'envoyer un bruit blanc (ou presque) et de voir comment réagissent les billes. Pour lors on ne s'intéressera qu'au mouvement d'une bille seule dans une grande forme, de sorte à minimiser les collisions (avec le bord).

Il faut faire attention à :

- la gamme de fréquences : Si on envoie des fréquences trop élevées le système ne suivra plus, si on envoie des fréquences trop faibles, le déplacement de la planche par rapport à la bille deviendra visible pour la caméra et il n'y aura plus de sens aux résultats. C'est pourquoi on choisit des fréquences entre 20 et 200 Hz.

- l'amplitude du voltage de la commande en tension. Là encore, rien ne sert d'envoyer un voltage trop élevé sans quoi le déplacement relatif de la planche par rapport à la bille est trop important. De même il ne faut pas descendre trop bas sans quoi la bille ne perçoit plus rien (l'agitation n'est pas suffisante pour dépasser l'effet des frottements solides).

L'amplitude du voltage est cependant le seul paramètre sur lequel il est pertinent de jouer. On l'appellera A dans la suite.

I.2 Comment caractériser un bain thermique ?

Répartition des vitesses La répartition des vitesses doit suivre une loi gaussienne centrée en 0. La vitesse moyenne de la bille est nulle, mais sa vitesse quadratique moyenne $u = \sqrt{\langle v^2 \rangle}$ ne l'est pas et est liée à T par la formule : $u = \sqrt{\dfrac{2k_B T_{eff}}{m}}$ car on a deux degrés de liberté, et où on note T_{eff} la température effective qu'on aurait si jamais on était en situation réelle.

On doit pouvoir vérifier en particulier la statistique de Maxwell-Boltzmann :

$$p(\vec{v}) = p_{v,0}.\exp\left(-\frac{m.v^2}{2k_B T_{eff}}\right)$$

où $p_{v,0}$ est une constante de normalisation et $p(\vec{v})$ la probabilité de trouver la particule à la vitesse $\vec{v}$.

Répartition des positions Lorsqu'on incline le support d'un petit angle α, on obtient un dénivelé $h(\alpha, x) \simeq x.\alpha$ en choisissant x comme la ligne de plus grande pente. Si le dispositif caractérise bien un bain thermique on devrait vérifier en particulier :

$$p(x) = p_{x,0}.\exp\left(-\frac{m.g.h(x,\alpha)}{k_B T_{eff}}\right)$$

où $p_{x,0}$ est une constante de normalisation et $p(x)$ la probabilité de trouver la particule en x.

I.3 Expérience 1 : Répartition des vitesses et loi d'évolution de T_{eff} en fonction de A

I.3.1 Descriptif de l'expérience

On fait varier la valeur de A sur les deux hauts parleurs et grâce à la caméra CCD on récupère toutes les 10 ms une image de la bille dans sa figure, après avoir ajusté l'horizontalité de la planche et l'éclairage.

Les images sont alors traitées par l'ordinateur afin de récupérer les trajectoires des particules en fonction du temps, puis leur vitesse instantanée, et de modéliser la répartition des vitesses obtenue.

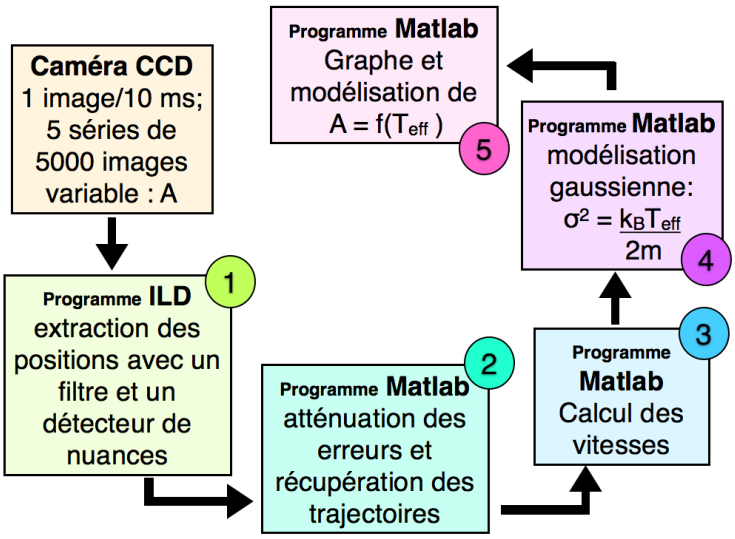

FIGURE 4 – Schéma de procédure d'exploitation des données pour la première expérience

I.3.2 Difficultés lors de l'exploitation des données

1 - programme IDL Nous avons récupéré le programme IDL qu'on nous avait donné pour détecter automatiquement la position de la bille sur les images. Toutefois, pour les quantités d'images que nous voulions exploiter (25000 images par acquisition) le traitement était particulièrement long.

Premier problème : **la détection de la bille** n'était pas optimale. En effet, si les rebords du disque dans lequel était emprisonnée la bille brillaient trop, l'ordinateur détectait plusieurs points, et ce de façon assez fréquente. De même lorsque la bille créait elle-même un reflet sur son support. Nous avons donc pris le temps de créer un filtre propre sur lequel on essayait d'enlever des bords trop lumineux, et nous avons dans le programme, modifié les paramètres de seuil afin que la détection se fasse principalement sur la bille.

Deuxième problème : il restait la difficulté du **temps d'exécution** du programme. (Au début environ une heure pour 25000 images). Nous avions remarqué que le programme ne cessait d'afficher à chaque fois qu'il détectait la bille sur une image, l'image en question et le (ou les) points retenus. De même s'il y avait une erreur (plusieurs points détectés), alors le programme s'arrêtait une fraction de seconde. (Quand il y avait en moyenne 10 % d'erreurs sur la détection, cela faisait beaucoup).

Nous avons donc déjà supprimé la première perte de temps en tentant de l'empêcher d'afficher les images à répétition. Ceci était une épreuve délicate, car nous ne connaissions pas le langage IDL et reprendre le programme d'une autre personne n'est pas évident.

2 - programme Matlab : atténuation des erreurs Le programme qu'on nous fournissait, qui donnait les trajectoires des billes, supprimait, pour les images sur lesquelles le nombre de points ne correspondaient pas au nombre de billes de l'expérience, la ligne de valeurs. A priori sous les hypothèses d'un petit nombre d'erreurs et d'une répartition aléatoire des erreurs, cela importait peu, mais nous avions trop d'erreurs pour considérer cela.

Lorsque sur l'image i l'éclairement sur la bille n'était pas optimal, comme nous l'avons dit ci-dessus, le programme récupérait plusieurs points. Aussi, avons nous ajouter sur le programme, dans ce cas là, un calcul des écarts entre chacun des points pris sur cette image par rapport au point sur l'image i-1. Nous avons ensuite adopté comme critère que le plus petit écart caractériserait la position de la bille sur l'image i.

Code 1 en annexe - page 33

3 - programme Matlab : calcul des vitesses Le mouvement étant à priori comparable selon toutes les directions horizontales si le réglage de l'horizontalité a été bien fait, on décide à partir de ce moment-là, et ce n'est pas choquant, de n'étudier le champ des vitesses que selon une direction (ici celle des x est fixée arbitrairement).

4 - programme Matlab : Détection gaussienne Nous avons compris que le programme Matlab qui nous était fourni choisissait un écart typique de vitesses : Δv et puis classait la vitesse obtenue pour chaque image i dans un intervalle du type : $[v; v + \Delta v]$ Ainsi on obtient un graphe de la probabilité que la bille ait une vitesse comprise dans un intervalle : $[v; v + \Delta v]$.

Mais ce critère était peu satisfaisant. En effet, à raison de diminuer Δv excessivement, on obtient un point par case, et la gaussienne est complètement raplatie. A raison d'augmenter Δv excessivement, on obtient une gaussienne élargie excessivement, car on n'a pas assez de points : on perd donc de la résolution. On pourrait se dire qu'un bon ordre de grandeur pour Δv est $\Delta v \sim \sqrt{\frac{2k_B T_{eff}}{m}}$ mais n'ayant pas accès à la température hypothétique, parce que c'est justement ce qu'on cherche, l'évaluation était impossible à priori.

Par conséquent, nous avons décidé de créer un autre programme pour une autre modélisation où cette imprécision serait supprimée et où on pourrait utiliser l'ensemble des points acquis sans compromis. Ainsi on a décidé de prendre l'intégrale de la gaussienne. Voilà le principe :

- On classe les vitesses et on s'intéresse au nombre d'images (rapporté au nombre total d'images, pour avoir une probabilité) sur lesquelles la vitesse de la bille est inférieure ou égale à v et on obtient une courbe expérimentale $p_{exp}(v)$
- On modélise le tout par l'intégrale de la gaussienne :

$$p_{th}(v) = \mathbb{P}(u \leq v) = \sqrt{\frac{m}{2\pi k_B T_{eff}}} \int_{-\infty}^{v} \exp\left(-\frac{mu^2}{2k_B T_{eff}}\right) du$$

On tente alors de définir un critère de fit, une grandeur χ qui doit être petite devant l'unité pour un bon fit. On pose donc

$$\chi = \frac{1}{\sqrt{\frac{2k_B T_{eff}}{m}}} \int_{-\infty}^{+\infty} (p_{exp}(v) - p_{th}(v))^2 dv$$

En revanche, les données expérimentales n'étaient connues qu'en un nombre (certes grand) fini de points, c'est pourquoi, au lieu d'effectuer une intégrale, nous avons approché l'intégrale par une somme de Riemann comme suit :

$$\sum_{i=1}^{N-1} (p_{exp}(v_i) - p_{th}(v_i))^2 . (v_{i+1} - v_i)$$

où on arrête la somme à $N-1$ comme pour toute somme de Riemann qui se respecte.

Il reste à normaliser pour avoir un critère de fit. Il s'agit de trouver un facteur d'erreur pour les probabilités, et un pour l'écart typique entre les vitesses.

• Pour les probabilités : l'erreur qu'on commet est de l'ordre de $\frac{1}{N}$ vu la définition prise pour les probabilités. On a alors un écart type de l'ordre de $\sigma_p(v_i) = \sqrt{\frac{p(v_i)}{N}}$ Or $p(v_i)$ caractérise le nombre de photos sur lesquelles la bille a une vitesse inférieure ou égale à v_i. Sur le domaine pertinent, cette probabilité est de l'ordre de 1. Donc $\sigma_p^2 \simeq \frac{1}{N}$

• Pour les vitesses. On pose $\Delta v_i = v_{i+1} - v_i$, et puis on cherche l'écart type $\sigma_{\Delta v}$ associé grâce à une simple modélisation gaussienne.

On pose alors $\chi = \frac{1}{\sigma_p^2 \sigma_{\Delta v}} \sum_{i=1}^{N-1} (p_{exp}(v_i) - p_{th}(v_i))^2 . (v_{i+1} - v_i)$; où chaque membre de la somme est de l'ordre de 1 donc il est pertinent de diviser la somme par : $\frac{1}{N-p}$ où p est le nombre de paramètres fixés (ici 2 sur 25000 expériences, soit largement négligeable devant N).

Donc on définit la variable réduite $\chi_r = \frac{1}{N-p}\chi$ qui pour un bon fit doit être au plus de l'ordre de 1, et si possible plus petite. Après un rapide calcul, finalement :

$$\chi_r = \frac{1}{\sigma_{\Delta v}} \sum_{i=1}^{N-1} (p_{exp}(v_i) - p_{th}(v_i))^2 . (v_{i+1} - v_i)$$

Code 2 en annexe - page 34

I.3.3 Résultats

Courbes d'évolution Voici l'un de nos graphiques montrant l'évolution de $p_{exp}(v)$ avec la modélisation, pour l'une des dix valeurs d'excitation prises, centrée sur la zone sur laquelle la grande majorité de nos points expérimentaux sont placés :

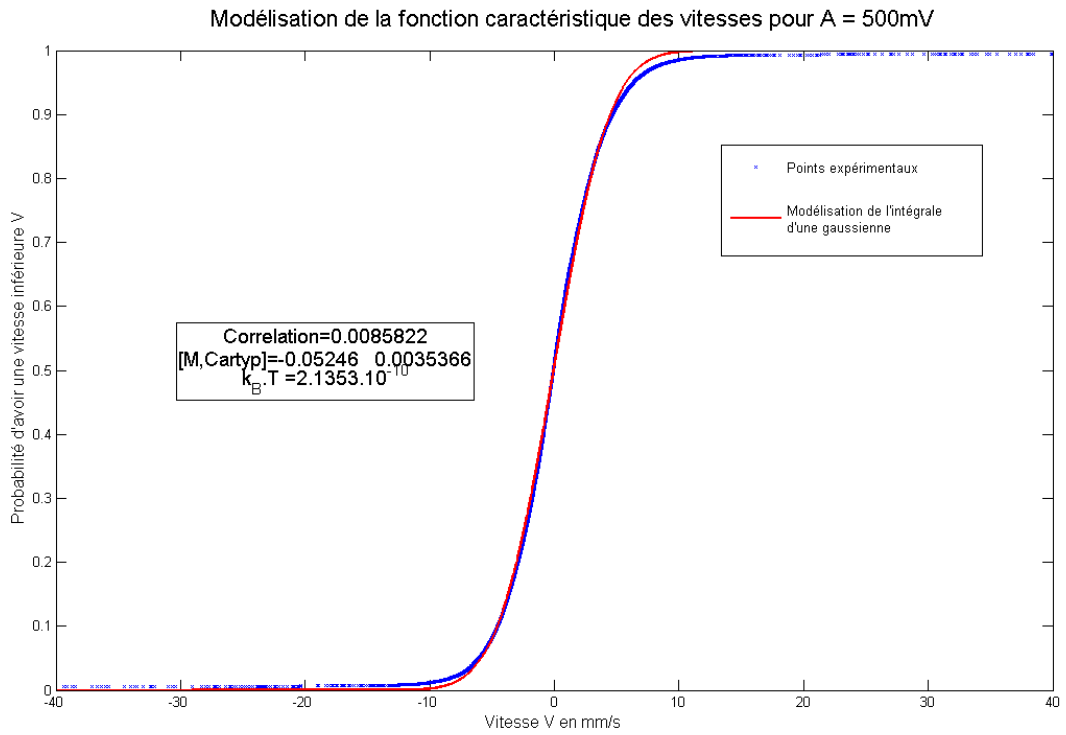

FIGURE 5 – Modélisation intégrale du profil des vitesses à une excitation 500 mV

Sur ce graphique, toutes les grandeurs numériques sont exprimées dans les unités du système international, et le coefficient de corrélation est ce qu'on a appelé χ_r, calculé en utilisant une modélisation par l'intégrale d'une gaussienne la répartition de l'écart typique entre les vitesses, comme suit.

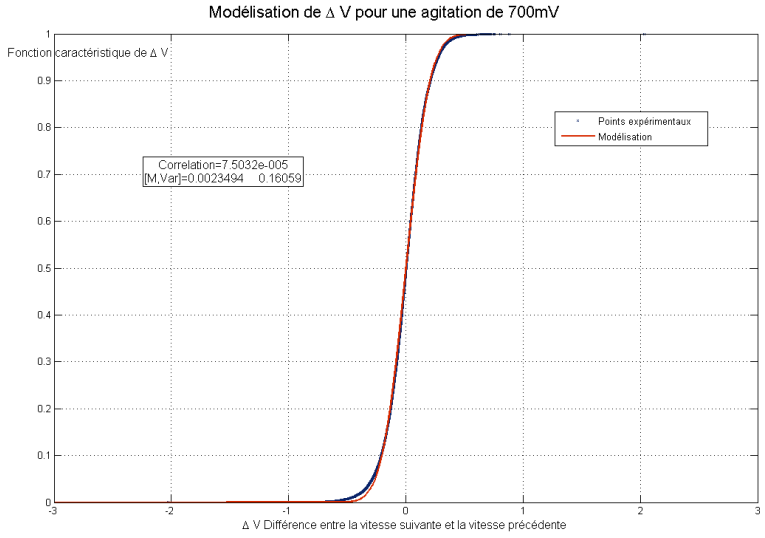

FIGURE 6 – Modélisation intégrale du profil de Δv à une excitation 700 mV

Pour les différentes excitations, nous avons bien sûr vérifié que l'écart type était comparable et avons sorti un $\sigma_{\Delta v} \simeq 0,2\ m.s^{-1}$ général.

Loi de variation de $T_{eff} = f(A)$

Traitement avec incertitudes

Nous avons décidé d'évaluer l'incertitude sur T_{eff} en calculant T_{eff} selon la même méthode que précédemment mais sur un nombre d'images et des images différentes. Comme nous n'avions pas beaucoup de temps, et en particulier pas le temps de coder un programme pour faire ça à notre place, nous avons évalué l'incertitude de façon sommaire en calculant T_{eff} sur cinq blocs de 5000 images.

Modèle à choisir

Il était certain que la croissance de A devait engendrer une croissance de T_{eff}. Mais comment caractériser cette croissance ?

En regardant l'allure de T_{eff} en fonction de A on aurait pu y caler un modèle linéaire. Toutefois, cela ne nous paraissait pas physiquement licite.

En effet $k_B T_{eff}$ représente une énergie tout comme $\frac{A^2}{R_{car}}$ où R_{car} est une résistance caractéristique associée au montage. D'où par analyse dimensionnelle, une loi d'évolution quadratique. Un autre argument consiste à s'intéresser aux frottements bille-plaque qui donnent une vitesse proportionnelle à l'amplitude des mouvements de la plaque. Ceci donne également une loi d'évolution quadratique.

C'est le modèle que nous avons essayé de superposer à nos points.

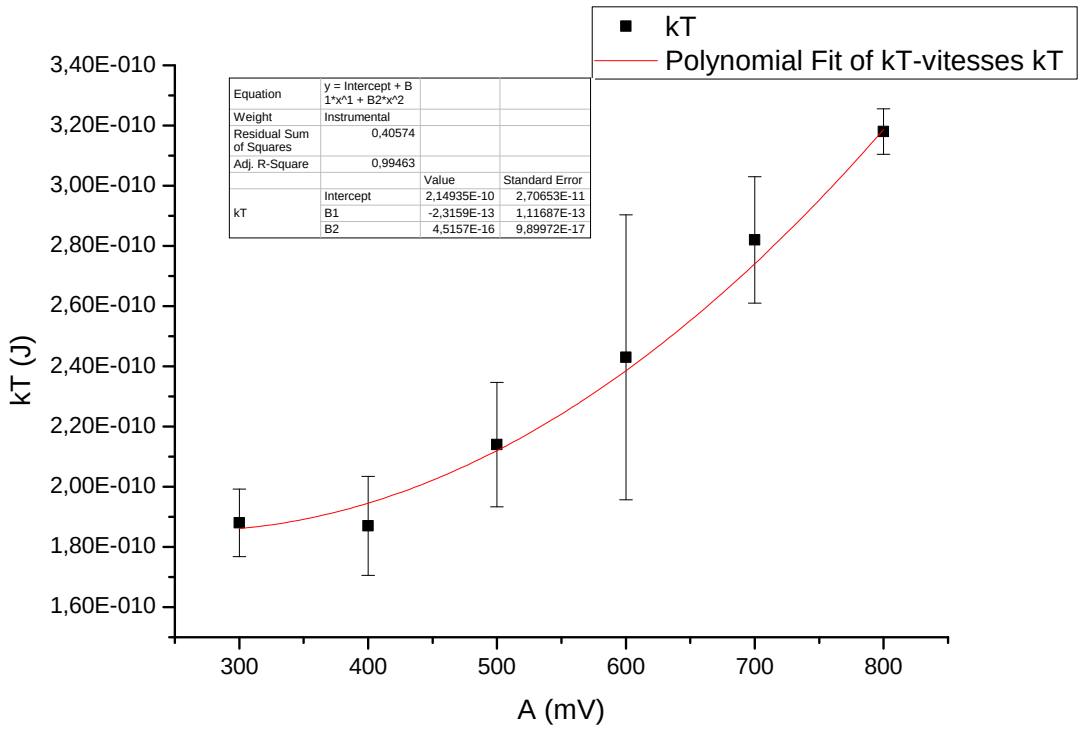

FIGURE 7 – Modélisation polynomiale (second degré) de l'évolution de $k_B T_{eff}$ en fonction de A

Remarque Il est à noté toutefois qu'à 600 mV les barres d'incertitude sont assez importantes ! (on verra par la suite que cette valeur d'agitation a toujours posé problème).

I.4 Expérience 2 : Répartition des positions et loi d'évolution de T_{eff} en fonction de A

I.4.1 Descriptif de l'expérience

On fait varier la valeur de A sur les deux hauts parleurs et grâce à la caméra CCD on récupère toutes les 10 ms une image de la bille dans sa figure, après avoir ajusté la plaque pour qu'elle fasse un petit angle α avec l'horizontale.

Les images sont traitées par l'ordinateur afin de récupérer les positions des particules en fonction du temps, et de modéliser la répartition des positions des particules.

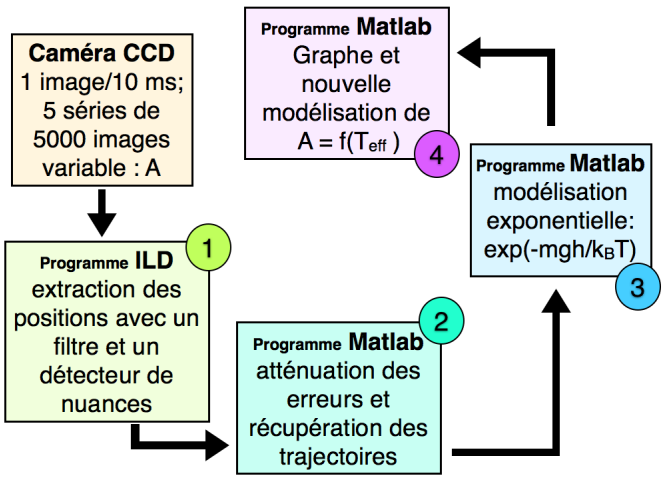

FIGURE 8 – Schéma de procédure d'exploitation des données pour la deuxième expérience

I.4.2 Difficultés lors de l'exploitation des données

Choix de la forme des bords Le problème n'est plus à symétrie sphérique, puisqu'on incline la plateforme selon une direction. Initialement nous avions laissé le cercle, mais nous nous sommes rendus compte que l'utilisation du cercle était ridicule. En effet, cela pouvait fausser les résultats de la manière suivante : pour un cercle il n'existe qu'un point correspondant à la position la plus "équilibrée", c'est-à-dire ayant l'altitude la plus basse. Pour un carré ou un rectangle incliné selon un de ses côtés, il en existe autant à n'importe quelle altitude.

Toutefois, il nous restait un regret quant à l'utilisation du carré : nous constations que le nombre de chocs sur le bord inférieur du carré était relativement conséquent. Nous pensons que cela a sans doute eu une incidence sur la qualité des résultats, mais finalement la courbe d'évolution obtenue a été tout à fait acceptable.

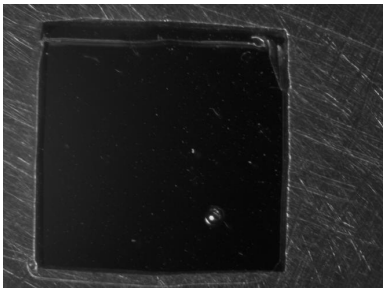

FIGURE 9 – Une bille dans le trou carré

3 - Matlab : modélisation exponentielle Le gros problème de la modélisation exponentielle sous Matlab est qu'elle ne converge pas. Nous avons donc modélisé le logarithme de nos probabilités (toujours une loi $p_{exp}(x)$ qui, comme pour les vitesses, caractérise la probabilité que la variable aléatoire X soit inférieure ou égale à x). Une fois cette modélisation réussie, nous avons lancé Matlab sur une modélisation exponentielle, en utilisant comme point de départ les valeurs des paramètres obtenus pour la modélisation linéaire.

La loi d'évolution théorique était la suivante :

$$p_{th}(x) = \mathbb{P}(X \leq x) = \exp\left(-\frac{m.g.(x - x_0).\alpha}{k_B T_{eff}}\right)$$

Remarques :

- l'introduction d'une origine était nécessaire (comme on va le voir sur les graphes ci-dessous).

En effet, la présence du bord inférieur du carré (en $x = 0$) entraîne que la courbe tend vers une pente nulle en $x = 0$ (c'est comme si on superposait deux exponentielles décroissantes : l'exponentielle réelle, et l'exponentielle miroir qui correspond aux chocs élastiques contre la paroi, au voisinage de 0 : d'où la pente nulle). Le comportement au voisinage de l'origine n'est donc pas pertinent, et il ne faut pas que la modélisation en tienne rigueur : d'où la nécessité d'introduire une origine.

- On a introduit un coefficient de corrélation exactement analogue à celui du modèle pour les vitesses.

Code 3 en annexe - 36

I.4.3 Résultats

Courbes d'évolution Voici l'un de nos graphiques montrant l'évolution de $p_{exp}(x)$ avec la modélisation, pour l'une des huit valeurs d'excitation prises. On observe bien la présence de ce palier significatif en $x = 0$.

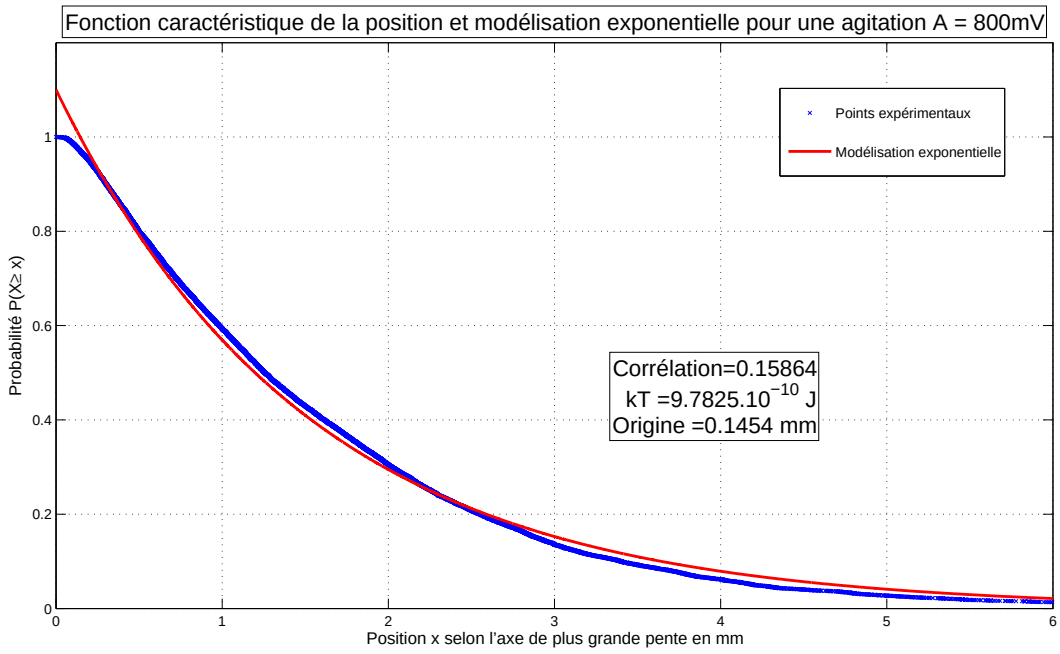

FIGURE 10 – Modélisation intégrale du profil des positions à une excitation 800 mV

Loi de variation de $T_{eff} = f(A)$

Nous avons procédé de la même manière pour évaluer l'incertitude sur T_{eff} et aussi proposé un modèle quadratique.

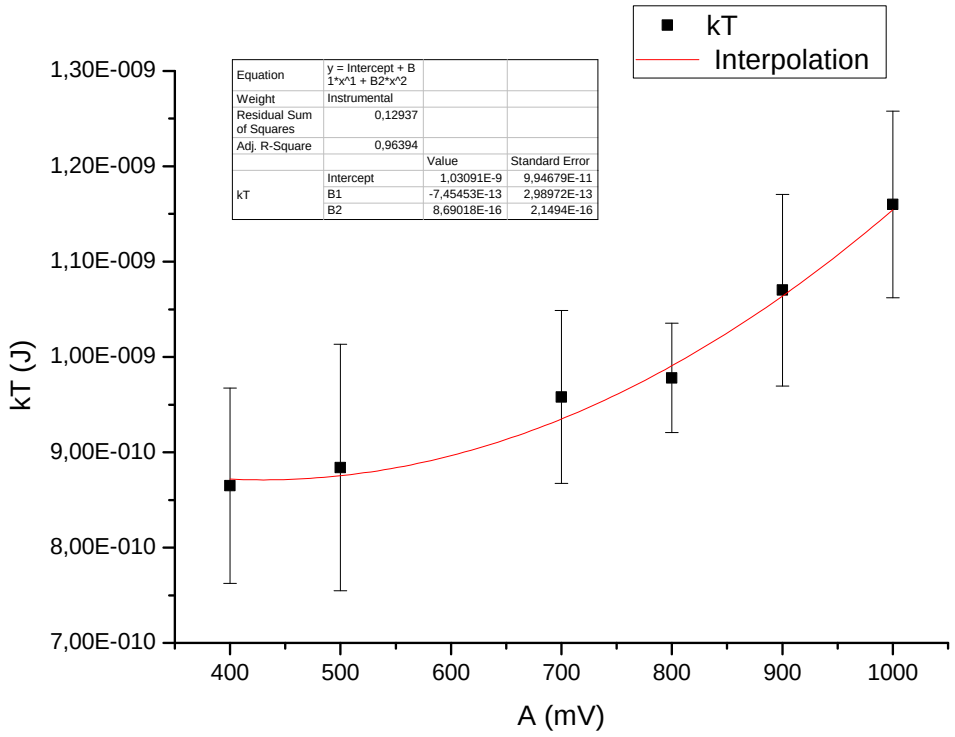

FIGURE 11 – Modélisation polynomiale (second degré) de l'évolution de k_BT en fonction de A

Remarque Ici nous avons carrément supprimé la valeur obtenue à 600 mV tant elle était incohérente. En effet, nous avons refait les mesures correspondantes, mais sans plus de succès.

I.5 Conclusion

Nous avons donc obtenu deux preuves que l'agitation des hauts parleurs incitait la bille à se comporter de façon semblable à un atome soumis à l'agitation thermique. De plus nous avons établi deux lois qui nous permettent de régler une température voulue à partir de l'agitation. Au terme de cette première partie, nous avons donc l'assurance d'avoir pour nos billes une bonne reproduction de l'agitation thermique. Il ne reste qu'à charger les billes avec le générateur de haute tension, et à voir comment elles se comportent quand on les met ensemble.

II Etude d'un cristal 2D à six billes

Nous allons maintenant étudier le comportement de six billes chargées dans un disque, en fonction de l'agitation thermique.

II.1 Présentation du dispositif et nouvelles observations

II.1.1 Dispositif : les Hexagones et les Pentagones

Le dispositif ne change pas, si ce n'est qu'on met en route le potentiel avec précaution, et qu'on étudie six billes au lieu d'une.

Avant de commencer les acquisitions nous avons regardé un peu à l'œil nu ce qu'il se passait. Au début, et la plupart du temps, les six billes s'arrangeaient en configuration pentagonale avec une bille au centre. On appellera cet état Pentagone dans la suite. Mais de temps en temps on voyait apparaître une configuration hexagonale. On appellera cet état Hexagone dans la suite.

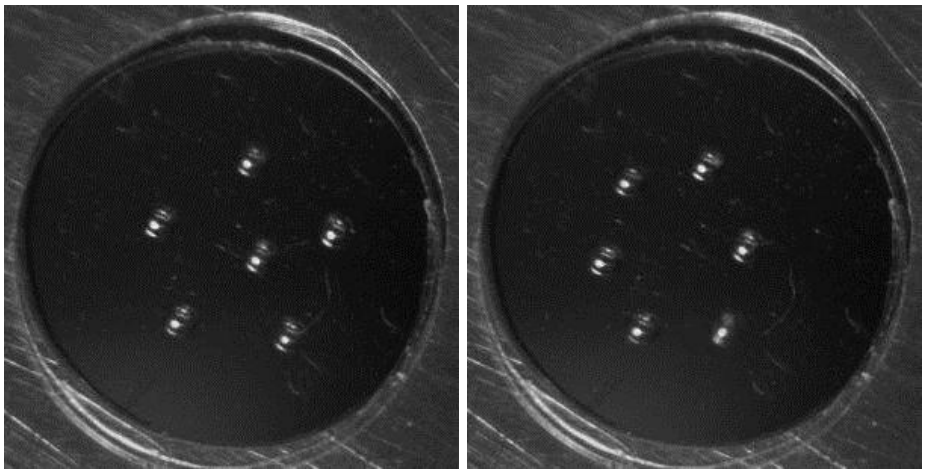

FIGURE 12 – Plus beaux spécimens d'hexagone et de pentagone. On a dû coder un programme annexe pour récupérer ces photos, tant les belles configurations sont rares !

Il était aussi facile de remarquer qu'à basse température les Hexagones apparaissaient peu, mais à haute beaucoup plus.

Nous avons donc établi la procédure de travail suivante :

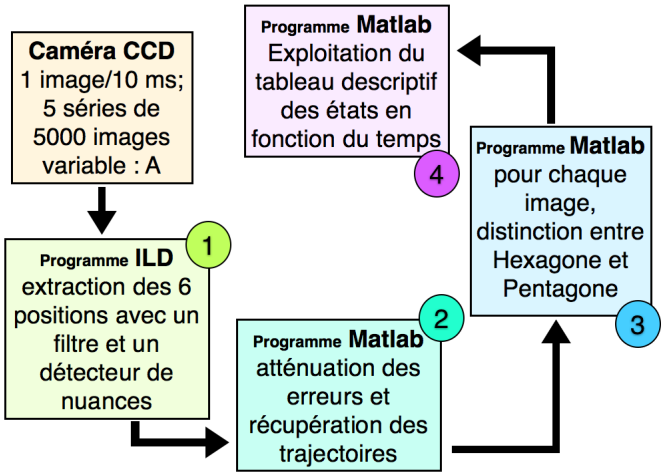

FIGURE 13 – Schéma de procédure d'exploitation des données pour l'expérience à six billes

II.1.2 Idées d'exploitation

Deux idées d'exploitation nous sont apparues très rapidement :

- Caractériser pour un grand nombre d'images à température donnée, le nombre d'images dans une configuration donnée. En effet, si l'on nomme E_H l'énergie d'un hexagone et E_P l'énergie d'un pentagone, la statistique de Maxwell Boltzmann postule que :

$$n_P(T) = n_0 \exp\left(\frac{-E_P}{k_B T}\right)$$

$$n_H(T) = n_0 \exp\left(\frac{-E_H}{k_B T}\right)$$

On peut faire cette analogie parce que compte tenu du nombre d'images, on a accès à une bonne approximation de la répartition statistique des états. Il n'est pas pertinent de considérer les corrélations entre un instant donné et l'instant d'après, car seule la répartition statistique nous intéresse finalement. Par conséquent les 25000 photos d'un même système à la suite sont comparables à la photo prise à un instant quelconque de 25000 sites indiscernables de 6 billes.
Dès lors le rapport de $n_P(T)$ et $n_H(T)$ donne accès à la différence d'énergie :

$$\boxed{E_P - E_H = k_B.T \ln\left(\frac{n_H(T)}{n_P(T)}\right)}$$

Où l'on a accès à T grâce aux lois déterminées précédemment.

- caractériser le palier d'énergie à franchir pour passer d'une configuration à l'autre.
On expose d'abord un petit point de chimie :
Soit la réaction $A \to B$ de constante de vitesse k. Alors la loi d'Eyring donne :

$$k = \frac{k_b T}{h} \exp\left(-\frac{\Delta_r G^{0,\neq}}{RT}\right)$$

où :
- la grandeur $\Delta_r G^{0,\neq}$ caractérise l'enthalpie libre de l'état de transition (état dans lequel, pour notre cas par exemple, la bille centrale du pentagone est légèrement excentrée)
- k est tel que $\frac{d[C_A]}{dt} = -k[C_A]$

Maintenant considérons qu'être dans un état Hexagonal ou Pentagonal détermine complètement l'état et qu'il n'y a donc pas de mémoire. En considérant que la loi d'Eyring est valable pour des molécules indépendantes, alors on peut considérer que ce que l'on a observé pour une molécule à des temps différents est la même chose que si l'on avait plein de molécules initialement dans le même état.
On adapte alors la loi d'Eyring à notre cas pour la réaction $H \to P$

$$k = \frac{k_B T}{h} \exp\left(-\frac{E^{\neq}}{k_B T}\right)$$

où :
- $E^{\neq}$ est l'énergie de l'état de transition
- $\frac{dN_H}{dt} = kN_H \Rightarrow N_H(t) = N_{H,0} \exp(-k.t)$

Conclusion : concrètement pour accéder à k on regarde combien de fois on a un groupe de configuration H (un groupe étant une suite continue de configurations H), on dit que ça correspond à $N_{H,0}$. Puis on trie les temps de vie des états H dans l'ordre croissant. Ensuite on retire 1 à $N_H(t)$ à chaque fois qu'on atteint un temps de vie observé. Et voilà on a une belle exponentielle. On mesure le temps caractéristique ($\frac{1}{k}$) par un fit exponentiel. Connaissant k_BT par nos expériences préparatoires, on accède à l'énergie de l'état de transition.

$$\boxed{E^{\neq} = k_B.T \ln\left(\frac{k_BT}{h.k}\right)}$$

Codes 4 en Annexe - page 38

II.2 Différenciation automatique entre un Hexagone et un Pentagone

La mise en œuvre de ces deux exploitations était impossible sans un premier travail préparatif pour détecter automatiquement, pour chaque image, si les billes sont en configuration H ou P. La détection devait remplir plusieurs critères : être efficace, et être d'une précision concordante avec celle de l'œil. En sortie, la fonction donnerait un grand tableau de 25000 lignes avec pour chaque ligne la lettre H ou P correspondant à la configuration.

De plus, certaines images, à l'œil, ne pouvaient être classées dans aucune des deux catégories. On nommera dans la suite ces configurations exotiques et intermédiaires par la lettre O. Elles peuvent être vues comme des états de transition. En voici quelques unes :

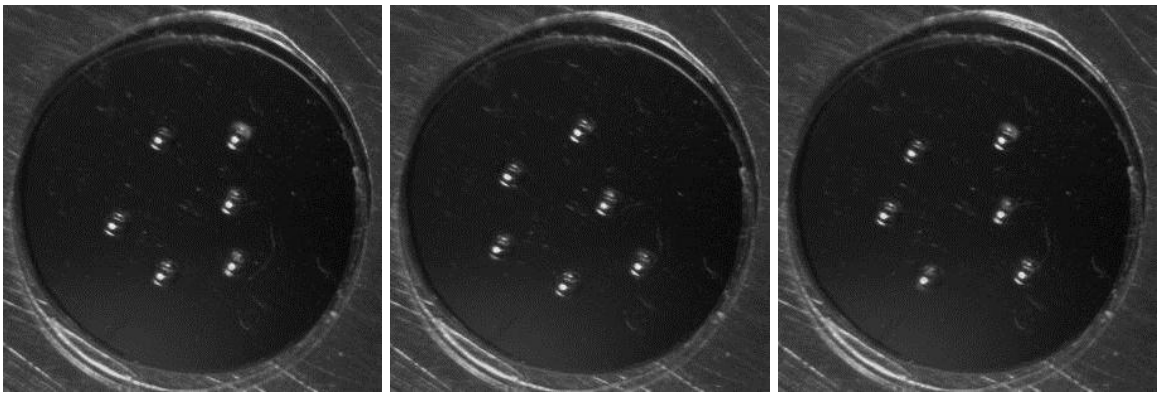

FIGURE 14 – Configurations indéterminées tant à l'oeil qu'automatiquement.

Il fallait donc :

- Trouver une méthode à priori précise et pratique de discrimination entre un Hexagone et un Pentagone (et sinon de classer l'image dans O).
- Optimiser cette méthode de détection de sorte à ce que l'œil soit d'accord avec les résultats donnés par l'ordinateur.

II.2.1 Recherche d'une méthode de différenciation

Idée 1 : Distance aux autres billes Notre premier programme s'intéressait à la distance moyenne aux autres billes.

Pour chaque image, on regarde la position des six billes :

- on calcule la distance moyenne de chaque bille aux 5 autres.
- on prend la plus grande et on lui soustrait la plus petite ce qui donne une variable notée d.
- on compare d aux d trouvés pour un Pentagone ou un Hexagone de référence, choisi à la main.

En effet, pour un Hexagone idéal, toutes les distances moyennes sont égales. Donc $d \simeq 0$.

Pour un Pentagone idéal en revanche, une des distances moyennes (celle de la bille centrale) est plus petite. Donc la différence $d \neq 0$.

La détermination était mauvaise, et mal concordante, notamment à cause de la difficulté de choisir un P et un H idéal et de fixer des seuils adéquats parce que la variable r était trop peu différente d'un Hexagone à un Pentagone.

Code 5 en Annexe -page 39

Idée 2 : Distance moyenne au barycentre Pour chaque image, on regarde la position des six billes :

- on calcule la position du barycentre
- on calcule la moyenne des distances au barycentre : m
- on compare m au m trouvé pour un Pentagone ou un Hexagone de référence.

Pour un H idéal : m est plus grand que pour un P idéal (car pour un pentagone l'une de ces distances est nulle).

La détermination était aussi mauvaise, toujours à cause de la détermination des configurations dites idéales.

Code 6 en Annexe - page 42

Idée 3 : Distance minimum et maximum au barycentre Suite à ces échecs nous avons résolu que la comparaison à des P ou H de référence n'était pas une bonne idée et qu'il nous fallait calculer une variable intrinsèque à l'image qui varie de 0 à 1, 0 pour H et 1 pour P ou l'inverse. Il était également peu approprié de calculer des valeurs moyennes. En effet moyenner a tendance à lisser les variations que l'on cherche à détecter.

Il nous a fallu changer juste un tout petit peu le programme précédent :

- on calcule la position du barycentre
- on calcule les distances de chaque bille au barycentre
- on pose $q = \dfrac{\text{plus petite distance au barycentre}}{\text{plus grande distance au barycentre}}$.

q remplissait alors toutes nos exigences :

- elle est toujours comprise entre 0 et 1.
- elle vaut 0 pour un Pentagone idéal, 1 pour Hexagone idéal.

Il ne restait plus qu'à fixer deux seuils :

- si $q \leq s_P$ alors l'image est P.
- si $q \geq s_H$ alors l'image est H.
- sinon elle est O.

Code 7 en annexe - page 44

II.2.2 Évaluation des seuils

Initialement, pour regarder la concordance de cette méthode avec la détection humaine, nous avions choisi $s_P = 0,25$ et $s_H = 0,75$.

Sur un bon millier de photos, la concordance était assez bonne. Mais nous ne pouvions pas tout vérifier à la main, et la détermination du moment où l'image était O était difficile. Parfois nous étions d'accord avec l'ordinateur, parfois non. Un tel seuil ne pouvait être qu'arbitraire (sur des configurations exotiques, la variable q pouvait se balader n'importe où).

Il fallait donc optimiser les seuils de sorte que quand l'on fait varier chaque seuil un petit peu le résultat varie peu. Si on observe que l'énergie dépend peu du choix des seuils dans un certain domaine, on sera certain de ne pas faire trop d'erreur dans le choix des seuils.

Optimisation d'un seuil unique Nous avons commencé par prendre $s_P = s_H = s$ et à tenter d'optimiser le seuil pris.

La démarche la plus simple à mener nous paraissait la suivante :

- on fait varier s entre 0 et 1 sur N valeurs.
- pour chaque valeur de s on fait tourner le programme de détection sur les 25000 images, et on calcule $N_{tot,H}$ et $N_{tot,P}$, nombre total d'images H, et d'images P. Puis on calcule : $r(s) = \ln(\frac{N_{tot,H}}{N_{tot,P}})$ car il est proportionnel à l'énergie et qui "modère" les variations trop brutales lorsqu'on arrive dans les zones peu pertinentes physiquement ($s \longrightarrow 0$, et $s \longrightarrow 1$). En plus de toutes ces qualités, c'est une grandeur qui conserve une certaine symétrie entre H et P.
- on trace le graphe $r(s)$ en fonction de s.
- on détermine à l'œil nu (parce que évaluer un minimum de dérivée sur quelque chose de potentiellement très irrégulier n'était pas pertinent) pour quelles valeurs de s, $r(s)$ varie peu.

Résultat

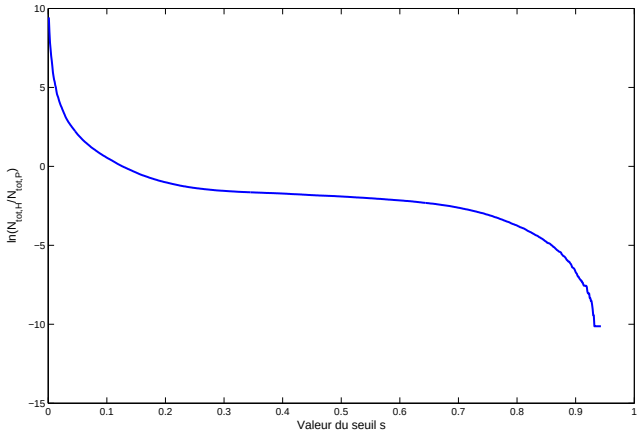

FIGURE 15 – Optimisation d'un seuil unique sur 1000 points de seuil

On constate que les valeurs de s pour lesquelles les valeurs relatives varient peu sont comprises environ pour $s \in [0.4; 0.6]$

Mais cela ne nous suffisait pas, car il était de toute façon plus pertinent de prendre deux seuils, et l'optimisation de deux seuils ne pouvait être bien plus complexe que celle d'un seul seuil.

Optimisation des deux seuils à la fois Une fois que nous avions réussi la première détermination nous avons décidé de réaliser la même chose en faisant varier à la fois s_P et s_H.

Le graphe obtenu serait alors un graphe en 3D, et après plusieurs débats, nous avons décider de retenir la même grandeur calculée, $r(s) = \ln(\frac{N_{tot,H}}{N_{tot,P}})$ car $E_P - E_H = k_B.T \ln\left(\frac{n_H(T)}{n_P(T)}\right)$.

Afin de faciliter ensuite la détermination du point correspond à un minimum de variation pour $E_P - E_H$, nous avons représenté en abscisse et en ordonnée non pas s_P et s_H mais la valeur moyenne des deux et l'écart entre les seuils.

Cette étude présenta de grandes difficultés de temps de calcul, mais il était plus efficace (pour nous, pas pour nos ordinateurs) d'attendre la nuit que les calculs se fassent plutôt que de s'embêter à trouver un code plus optimal (chose pas forcément possible). La programmation elle ne présenta pas de réelle difficulté si ce n'est dans l'obtention d'un beau graphique 3D en couleur.

Code 8 en Annexe - page 46

Résultat

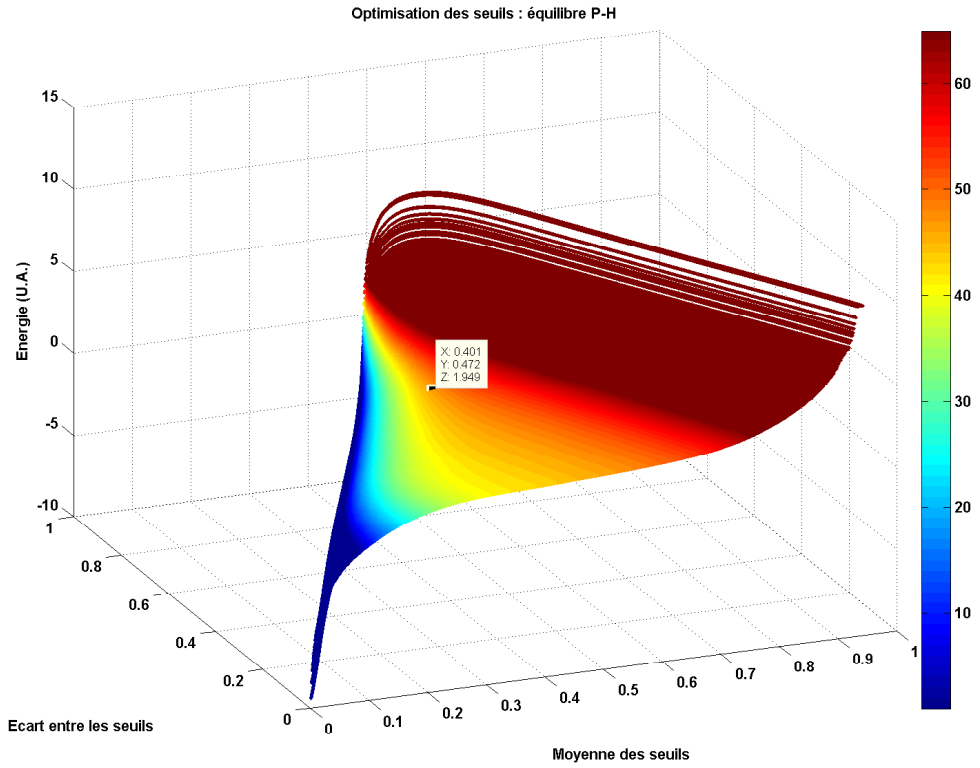

FIGURE 16 – Evolution de l'énergie en fonction des seuils et détermination du seuil optimal

La détermination du double seuil optimal s'est elle aussi faite à l'œil nu selon la procédure suivante :

- nous avons cherché sur la vue de profil (selon l'axe de moyenne des deux seuils) une valeur qui corresponde au centre de pente minimale.
- puis selon l'autre axe nous avons augmenté le plus l'écart entre les seuils.

On obtient finalement :

$$\boxed{\begin{cases} s_H = 0,637 \\ s_P = 0,165 \end{cases}}$$

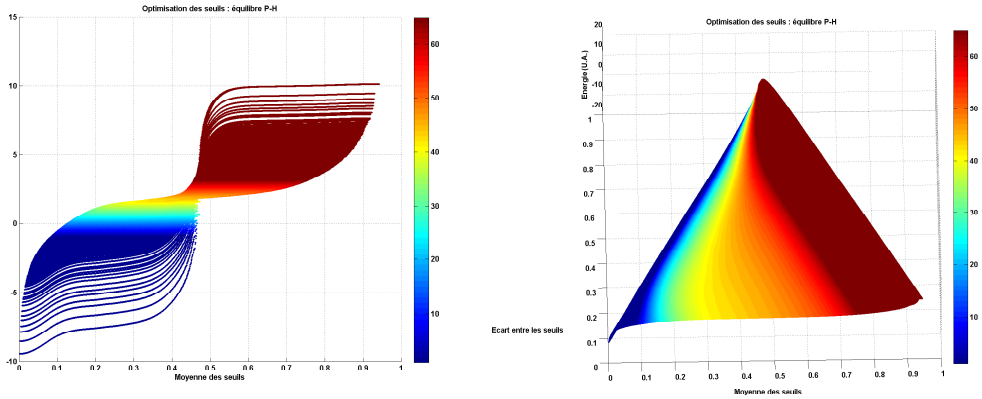

FIGURE 17 – Autres vues (de profil et de dessus) de l'évolution de l'énergie en fonction des seuils

II.3 Avant l'obtention des résultats : thermomètre

Un problème se posait évidemment pour déterminer toutes ces énergies : celui de la détermination de la température. Bien sûr, on pouvait utiliser l'une des deux lois établies en partie I. Cependant, ces expériences ayant été réalisées dans des conditions différentes (potentiel), on a décidé d'établir un thermomètre lié à l'évolution de nos six billes, de sorte que la mesure de la température soit indépendante de l'excitation qu'on met (parce que l'amplification du bruit est mal contrôlée).

Par conséquent, on a capturé avec la caméra sur la même image des six billes, une image avec une seule bille dans un petit disque. On récupère alors la température en étudiant la répartition moyenne des vitesses de la bille, par la même procédure développée en première partie. Et voilà un beau thermomètre !

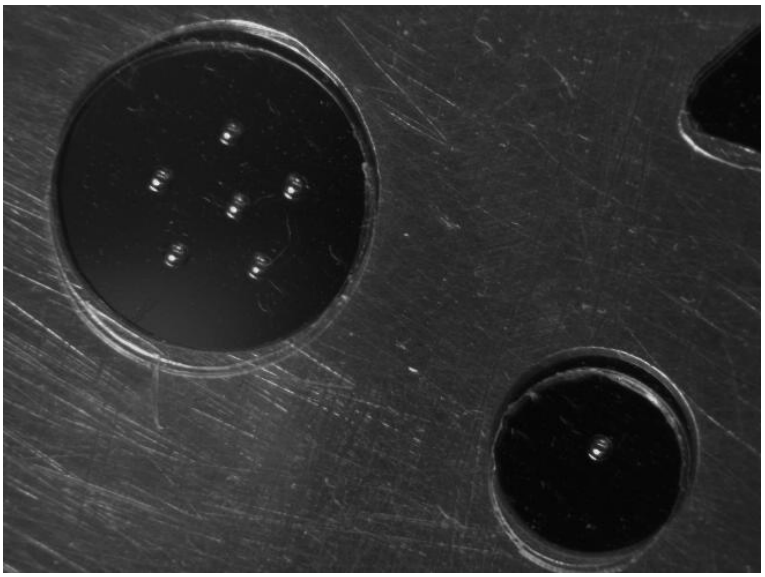

FIGURE 18 – Photographie de la bille thermomètre à côté des six billes dans le disque principal

II.4 Energie relative entre un Hexagone et un Pentagone

Il n'y avait donc plus qu'à exécuter le traitement des données selon la procédure établie ci-dessus avec les valeurs optimisées pour les seuils pour obtenir l'énergie relative entre un Hexagone et un Pentagone, tout en utilisant notre thermomètre.

Traitement d'incertitudes et défauts Nous avons effectué le même traitement d'incertitudes que celui développé dans la première partie. Il fallait aussi vérifier que la quantité d'images indéterminées n'excédait pas une valeur inacceptable, afin d'avoir suffisamment d'états P et H pour pouvoir faire de la statistique. Généralement, il y avait entre 1% et 10% d'interminées. Cependant, à 600 mV, nous avons noté que près de 24000 images sur 25000 étaient en configuration indéterminée, et c'était en accord avec les images. L'analyse informatisée ne donnait d'ailleurs que 3 images Pentagone, configuration normalement la plus stable, pour 250 Hexagone. Nous proposons face à ce résultat relativement aberrant, d'abord de supprimer les données acquises, et ensuite de l'interpréter par une probable et inattendue résonance de tout le système à 600 mV.

Résultats On obtient cependant une courbe absolument satisfaisante, qui nous donne une valeur du gap énergétique entre les deux états Pentagone / Hexagone :

$$\boxed{E_P - E_H \simeq 1,4.10^{-10} J}$$

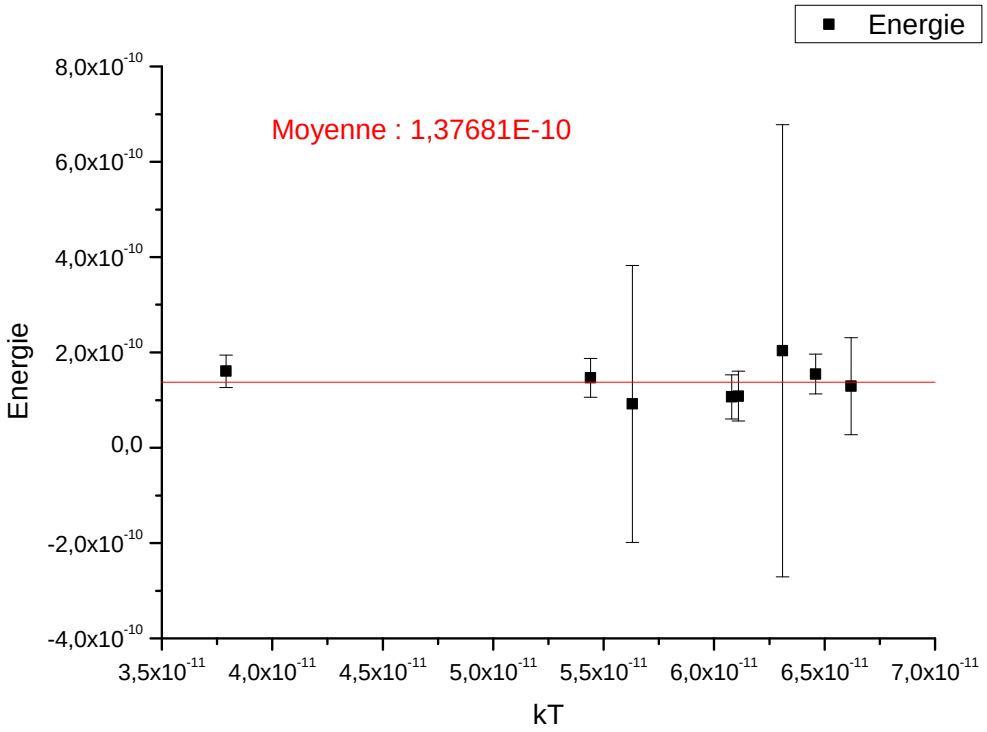

FIGURE 19 – Différence d'énergie entre les deux états stables

On remarque de plus que cette énergie est en ordre de grandeur dix fois supérieure à l'énergie thermique disponible, ce qui est cohérent : il est donc à priori difficile d'accéder à l'état Hexagone. L'étude suivante, qui montre l'énergie d'activation à fournir pour s'échapper de minimum relatif d'énergie correspondant à l'état Hexagone, confirme la difficulté à rester dans à cet état.

II.5 Durée de vie d'un état et énergie de l'état de transition

On a exécuté également le traitement des données ici selon la procédure établie plus haut. Nous avons eu une bonne surprise en regardant les résultats ! On a décidé de représenter l'évolution de la probabilité de survie d'une configuration hexagonale en fonction du temps, et on voyait directement une belle exponentielle. Cependant, la relative grande incertitude horizontale (0,04 s correspondant au temps entre chaque image) nous a poussés (comme elle n'était pas beaucoup plus petite que le temps caractéristique de décroissance) à rajouter une constante d'origine temporelle, qui dans nos modélisations est notée C, de telle sorte que : $\mathbb{P}(H \text{ survit a } t) = \exp(-k(t-C))$. Cette constante est toujours demeurée inférieure au temps d'incertitude horizontale donc elle corrigeait bien l'indétermination sur l'origine des temps.

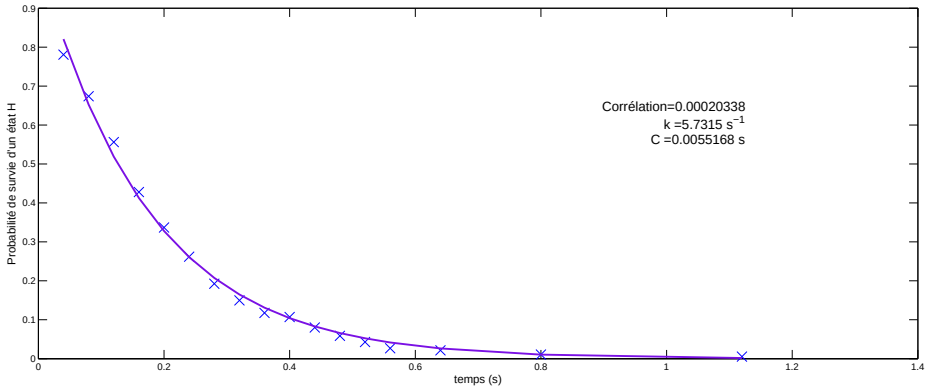

FIGURE 20 – Évolution de la probabilité de survie d'un Hexagone au cours du temps et modélisation

Défauts et améliorations Là encore les résultats n'étaient pas du tout cohérent à 600 mV.

De plus, le traitement des incertitudes fut plus complexe compte tenu du petit nombre de valeurs que nous obtenions correspondant au nombre de suites dans la configuration H (une petite trentaine, ce qui réparti en 5 séries donnait une demie douzaine de valeurs sur lesquelles déterminer un temps de décroissance). Nous avons donc exécuté la procédure habituelle quand cela était possible, et sinon nous avons propagé les incertitudes.

Enfin, la modélisation ne pouvant être très précise, nous avons décidé de négliger la contribution du terme de température en facteur de l'exponentielle pour $E^{\neq}$ (ce qui revient à ne considérer que la loi d'Arrhénius). En effet, à haute température, il existe potentiellement de plus en plus d'états correspondants à des états de transition, et ces états sont d'énergie en moyenne plus haute. Ce qui explique qu'une loi du type Arrhénius était finalement mieux adaptée (mois sensible à ce genre de choses).

Résultat Voilà la courbe que nous obtenons :

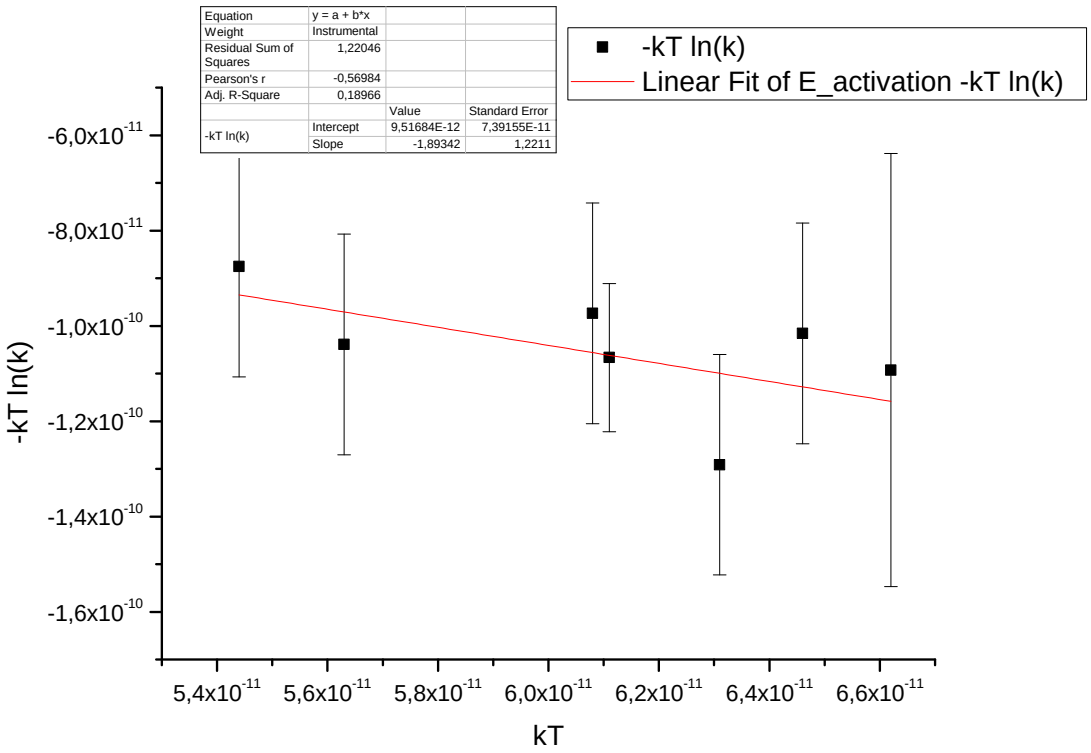

FIGURE 21 – Évolution de l'énergie d'activation pour passer de l'état H à l'état P en fonction de $k_B T$

On trouve :

$$\boxed{E^{\neq} \simeq 9.10^{-11} J}$$

Là encore c'est rassurant. Il faut à peu près la moitié de l'énergie entre H et P pour passer de l'état H à l'état de transition : le système ne restera pas bien longtemps dans l'état H.

Et dans l'autre sens ? Nous avons évidemment émis la suggestion qu'il serait bien d'effectuer le même traitement pour les temps de vie des états P et ainsi en déduire l'énergie d'activation pour passer de l'état P à l'état H, pour vérifier que le chemin serait à peu près le même. Néanmoins, nous avons manqué de temps.

II.6 Conclusion

Finalement nous sommes parvenus à modéliser le comportement statistique des « atomes »de ce cristal. En effet, nous avons maintenant une idée du comportement d'un Hexagone ou d'un Pentagone, de leur durée de vie, de leur énergie relative, nous sommes ainsi capable de faire des prévisions statistiques sur notre cristal. Mais pour mieux comprendre la stabilité relative des états Hexagone ou Pentagone, nous avons voulu étudier alors le potentiel d'interaction entre les billes.

III Étude du potentiel d'interaction

Ce qui nous intéresse principalement ici est de déterminer le potentiel d'interaction entre deux billes. Mais en condition réelle, les billes sont piégées dans un potentiel créé par le bord du cercle dans lequel elles ont été placées. Ainsi la statistique de répartition des billes est à la fois régie par l'interaction bille-bille et les interactions bille-bord. Le hamiltonien du système est composé de trois termes. Afin d'éliminer les deux termes dont on ne veut pas, nous allons commencer par déterminer l'énergie d'interaction entre une bille et le bord.

III.1 Potentiel créé par les bords

III.1.1 Dispositif expérimental

On utilise toujours le même dispositif. Ici le trou était circulaire, la tension imposée de 1000 V, l'agitation était à 800 mV.

(En réalité cette expérience avait déjà été réalisée en parallèle des mesures d'énergie dans le trou à six billes pour servir de « thermomètre » .)

III.1.2 Principe de l'analyse statistique

On étudie une bille ponctuelle évoluant dans un potentiel statique.

Définitions L'espace de probabilité devrait être l'espace des phases à quatre dimensions (2 spatiales et 2 en impulsions) équipé de la mesure de probabilité ayant pour densité $A.\mathrm{e}^{-\frac{E_{mécanique}}{kT}}$.

Le hamiltonien est une somme de termes purement impulsionnels et de termes purement spatiaux. On peut donc séparer ces parties. Ici seule la position nous intéresse ; on va donc se séparer de l'impulsion. Ainsi on retient le choix suivant :

L'espace de probabilité Ω est le disque (purement spatial) centré sur le centre du potentiel et de rayon égal à la distance du centre au point expérimental le plus éloigné (cette restriction est utile aux questions de normalisation. On pourrait en fait conserver toute la partie spatiale de l'espace des phases à deux dimensions). La mesure de probabilité est une mesure de densité $\rho(\vec{r}) = A.\mathrm{e}^{-\frac{V(\vec{r})}{kT}}$, avec $V(\vec{r})$ le potentiel en $\vec{r}$.

Soit la famille de fonctions $f_R : \vec{r} \mapsto \mathbb{1}_{\{||\vec{r}||<R\}}$.

Grandeur calculée et limite statistique Soit $N \in \mathbb{N}$ et $R \in \mathbb{R}^+$

Soit $\chi_R = \sum_{i=1}^{N} \frac{f_R(\vec{r}_i)}{N}$ avec $(\vec{r}_i)_{i\in\mathbb{N}}$ suite de tirage indépendants de points dans l'espace Ω.

D'après la loi des grands nombres,

$$\lim_{N\to\infty} \chi_R =< f_R >= \int f_R(\vec{r})\rho(\vec{r})\mathrm{d}\vec{r} = \int_0^R 2\pi.A.r.\mathrm{e}^{-\frac{V(r)}{kT}}\mathrm{d}r = \mathbb{P}(||\vec{r}|| < R)$$

$$\lim_{N\to\infty} \chi_R = \mathbb{P}(||\vec{r}|| < R) \tag{1}$$

Conclusion On va calculer la valeur expérimentale de χ_R. Puis dans l'approximation où N est très grand, on va l'identifier à $\mathbb{P}(r < R)$. Puis on dérive par rapport à R : $\frac{\mathrm{d}\chi_R}{\mathrm{d}R} = A.R.\mathrm{e}^{-\frac{V(R)}{kT}}$

Et ainsi $-ln(\frac{\mathrm{d}\chi_R}{\mathrm{d}R}) + ln(R) = Cste + \frac{V(R)}{kT}$.

Ainsi on peut donc obtenir le profil de potentiel à partir de la statistique de présence dans le puits. C'était l'objectif de la section. On obtient tout à une constante additive et une constante multiplicative près. Comme ici c'est la forme des potentiels qui nous intéresse, on ne cherchera pas à les déterminer. Il reste à voir la mise en pratique et les résultats.

III.1.3 Mise en pratique : les solutions retenues

Pour détecter la position de la bille, on a utilisé les techniques qui sont désormais habituelles.

Ensuite pour déterminer le centre du potentiel, la méthode retenue a été de calculer la position du barycentre des points de présence de la bille.

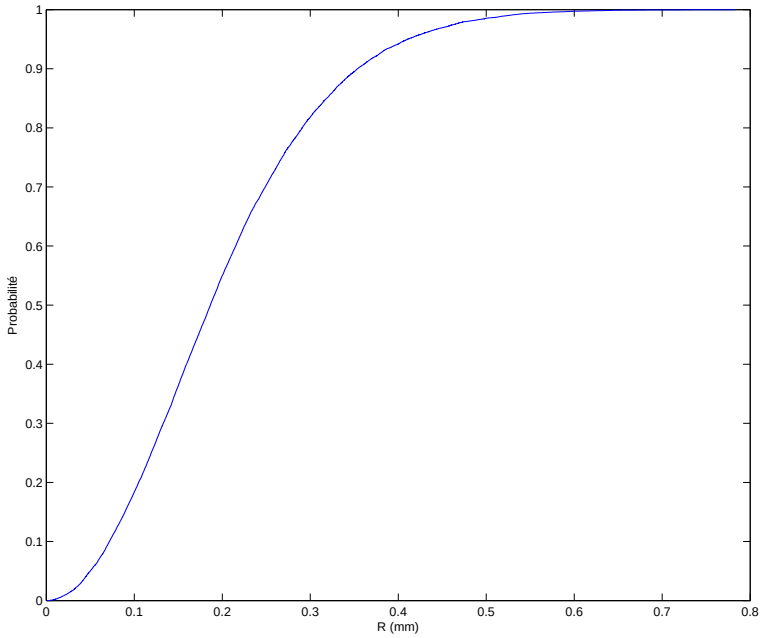

FIGURE 22 – Fonction caractéristique de la probabilité radiale de présence d'une bille par rapport à la distance au centre

Le point délicat est la dérivation numérique. En calculant le taux d'accroissement entre chaque point ce n'était pas satisfaisant.

Il a donc fallu trouver mieux.

On a donc exploré différentes manières de dériver numériquement :

- Réduction du nombre de points à écart fixe sur l'axe des ordonnées puis prise du taux d'accroissement.
- Réduction du nombre de points à écart fixe sur l'axe des abscisses puis prise du taux d'accroissement.
- Utilisation d'une fenêtre glissante à écart fixe sur l'axe des ordonnées et prise du taux d'accroissement entre les extrémités de la fenêtre.
- Utilisation d'une fenêtre glissante à écart fixe sur l'axe des abscisses et prise du taux d'accroissement entre les extrémités de la fenêtre.
- Interpolation polynômiale puis dérivée formelle du polynôme. Il s'est avéré nécessaire de réduire le nombre de points.
- Interpolation par une suite de polynômes du $3^{\text{ème}}$ degrés raccordés de manière C^1 puis dérivation formelle par morceau. Il s'est avéré nécessaire de réduire le nombre de points.

Comme attendu, l'interpolation polynômiale de Lagrange présente des comportements fantaisistes entre les points. L'interpolation par une suite de polynômes donne un lissage artificiel qui ne donne pas de meilleurs résultats que les autres techniques pourtant plus basiques.

La solution retenue a été celle de l'utilisation d'une fenêtre glissante horizontale.

Code 9 en Annexe -page 49

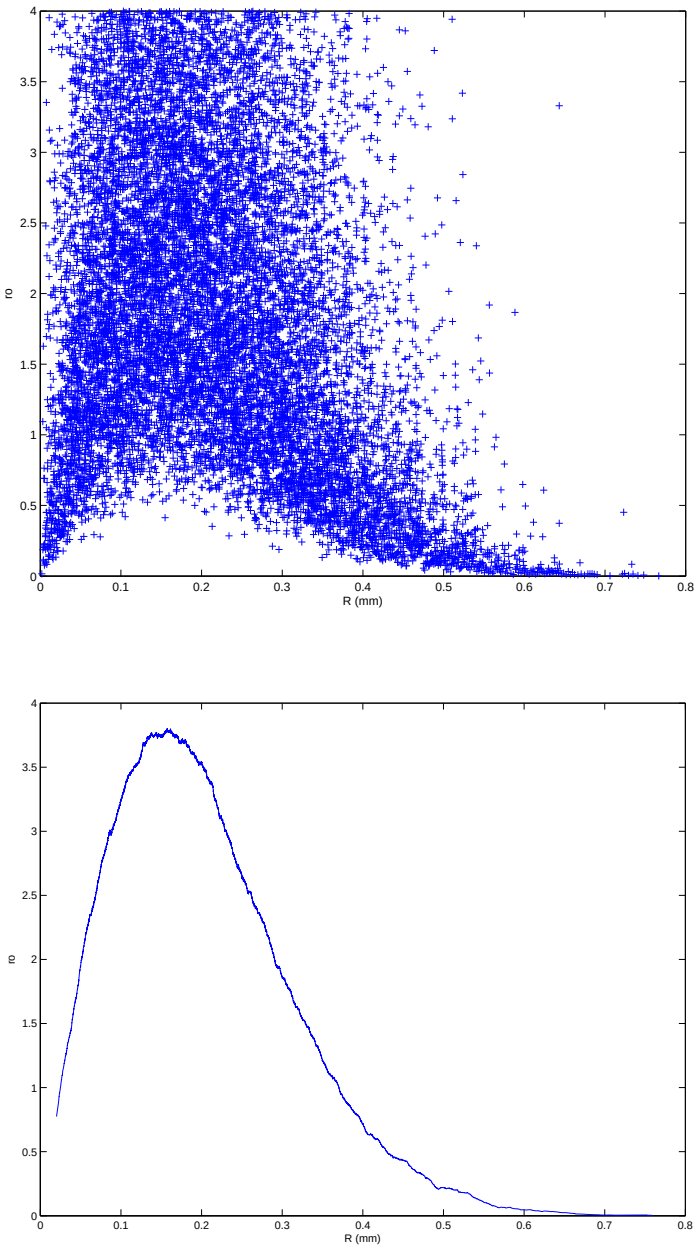

FIGURE 23 – Densité de probabilité radiale de présence d'une bille par rapport à la distance au centre, dérivation par taux d'accroissement simple (en haut) et avec fenêtre horizontal mobile (en bas)

III.1.4 Résultats

Pour l'usage ultérieur de ce potentiel, il s'est avéré utile de connaitre le potentiel pour des points éloignés de 1,6 mm alors que les points expérimentaux ne vont que jusqu'à 0,75 mm. Il c'est donc avéré nécessaire de procéder à une extrapolation.

Vu la forme du potentiel, il était judicieux de le modéliser par un polynôme du second degré. La symétrie de révolution du puits devait imposer au potentiel d'être nul à l'ordre 1. En imposant ou non cette condition, on obtenait des résultats peu différents. On a décidé de garder l'ordre 1 pour la suite.

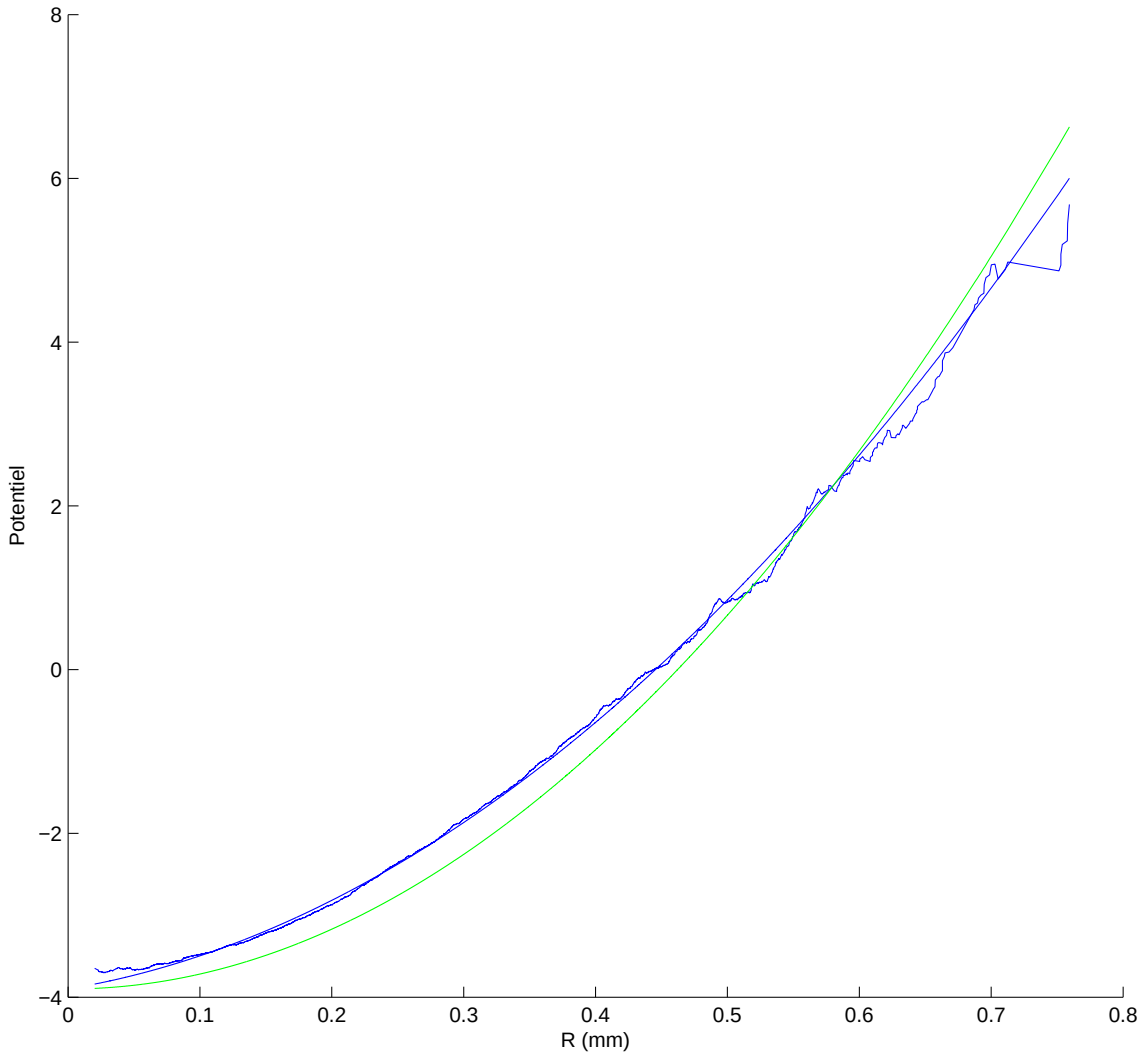

FIGURE 24 – Potentiel d'interaction bille-bord. En vert la courbe et la régression sans 1er degrés ; en bleu la régression avec 1er degrés ; en bleu également la courbe expérimentale.

III.2 Potentiel d'interaction à deux billes

III.2.1 Dispositif expérimental

Dans le même trou que celui qui a servi pour la détermination du potentiel créé par le bord, on a placé deux billes. Les tensions et agitations ont été choisies identiques à celles utilisées avec une seule bille.

III.2.2 Principe de l'analyse statistique

On étudie deux billes ponctuelles dans un potentiel statique.

Définitions L'espace de probabilité Ω est l'ensemble des points de $(\mathbb{R}^2)\times(\mathbb{R}^2)$ limité à l'espace utile (c'est-à-dire borné).
La mesure de probabilité admet pour densité $\rho(\vec{r}_1,\vec{r}_2) = A.\mathrm{e}^{-\frac{V(\vec{r}_1)}{kT}-\frac{V(\vec{r}_2)}{kT}-\frac{V_{1\leftrightarrow 2}(||\vec{r}_1-\vec{r}_2||)}{kT}}$

Soit la famille de fonctions $f_R : (\vec{r}_1,\vec{r}_2) \mapsto \mathrm{e}^{+\frac{V(\vec{r}_1)}{kT}+\frac{V(\vec{r}_2)}{kT}}.\mathbb{1}_{\{||\vec{r}_1-\vec{r}_2||<R\}}$

Grandeur calculée et limite statistique Soit $N\in\mathbb{N}$ et $R\in\mathbb{R}^+$

Soit $\chi_R = \sum_{i=1}^{N}\frac{f_R(\vec{r}_{1,i},\vec{r}_{2,i})}{N}$ avec $(\vec{r}_{1,i},\vec{r}_{2,i})_{i\in\mathbb{N}^2}$ suite de tirage indépendants de points dans l'espace Ω.

D'après la loi des grands nombres :

$$\begin{aligned}
\lim_{N\to\infty}\chi_R &= \int f_R(\vec{r}_1,\vec{r}_2).\rho(\vec{r}_1,\vec{r}_2).\mathrm{d}\vec{r}_1.\mathrm{d}\vec{r}_2 \\
&= \int\limits_{\{||\vec{r}_1-\vec{r}_2||<R_{max}\}}\ \int\limits_{\{\vec{r}_1+\vec{r}_2\in Compact\}} f_R(\vec{r}_1,\vec{r}_2).\rho(\vec{r}_1,\vec{r}_2).\mathrm{d}(\vec{r}_1-\vec{r}_2).\mathrm{d}(\frac{\vec{r}_1+\vec{r}_2}{2}) \\
&= \int\limits_{\{||\vec{r}_1-\vec{r}_2||<R_{max}\}}\ \int\limits_{\{\vec{r}_1+\vec{r}_2\in Compact\}} Cste.\mathbb{1}_{\{||\vec{r}_1-\vec{r}_2||<R\}}.\mathrm{e}^{-\frac{V_{1\leftrightarrow 2}(||\vec{r}_1-\vec{r}_2||)}{kT}}.\mathrm{d}(\vec{r}_1-\vec{r}_2).\mathrm{d}(\frac{\vec{r}_1+\vec{r}_2}{2}) \\
&= Cste.\int\limits_{\{||\vec{r}_1-\vec{r}_2||<R\}} r.\mathrm{e}^{-\frac{V_{1\leftrightarrow 2}(r)}{kT}}\mathrm{d}r \\
&= \mathbb{P}(||\vec{r}_1-\vec{r}_2||<R)
\end{aligned}$$

Conclusion On va calculer la valeur expérimentale de $\chi(R)\approx\mathbb{P}(||\vec{r}_1-\vec{r}_2||<R)$
Puis on dérive par rapport à R :
$\frac{\mathrm{d}\chi_R}{\mathrm{d}R} = A.R.\mathrm{e}^{-\frac{V_{1\leftrightarrow 2}(R)}{kT}}$
Et ainsi $-ln(\frac{\mathrm{d}\chi_R}{\mathrm{d}R})+ln(R) = Constante+\frac{V_{1\leftrightarrow 2}(R)}{kT}$.
On a donc accès à $V_{1\leftrightarrow 2}(R)$

III.2.3 Mise en pratique

Les calculs sont quasiment les mêmes que pour déterminer le potentiel d'une bille. La principale différence est qu'au lieu de simplement compter un nombre de cas, on rajoute un coefficient qui provient de l'expérience avec une seule bille. Il est nécessaire que la température utilisée soit la même dans les deux expériences.

Au début on a essayé de n'utiliser pour le potentiel bord-bille que des point expérimentaux (pas de régression). Le problème est que la répulsion entre les billes fait que les distances au centre du potentiel sont plus importantes avec deux billes qu'avec une seule. Ainsi certains points sortent de la zone qui a été explorée par la bille de la première expérience. Une première solution a été de simplement supprimer les points que l'on ne peut pas traiter. Mais en y réfléchissant, on s'est rendu compte que cela fausse la statistique (la modification non contrôlée du domaine d'intégration empêche la séparation des variables barycentre/distance).

Il a donc fallu trouver une autre solution. On a choisi d'utiliser une régression polynomiale du second degré pour le potentiel bord-bille.

La densité radiale de probabilité présente de fortes variations, cela provient de l'exponentielle. Mais ce qui nous intéresse est l'énergie (et donc le logarithme). La grande difficulté provient du fait que presque tout le poids statistique est détenu par quelques points particulièrement éloignés. Ainsi quel que soit N, au delà d'un certain rayon la condition $N >> 1$ n'est pas vérifiée. Mais il n'est pas évident d'éliminer certains points sans fausser la statistique. Le problème nécessiterait une approche mathématique qui sortirait du cadre d'un projet expérimental.

Cela rend la qualité du résultat assez sensible au paramètre de dérivation. On a donc créé un programme annexe qui effectue une boucle sur le paramètre de dérivation. On a conservé le paramètre donnant (visuellement) le meilleur compromis.

Code 10 en Annexe -page 52

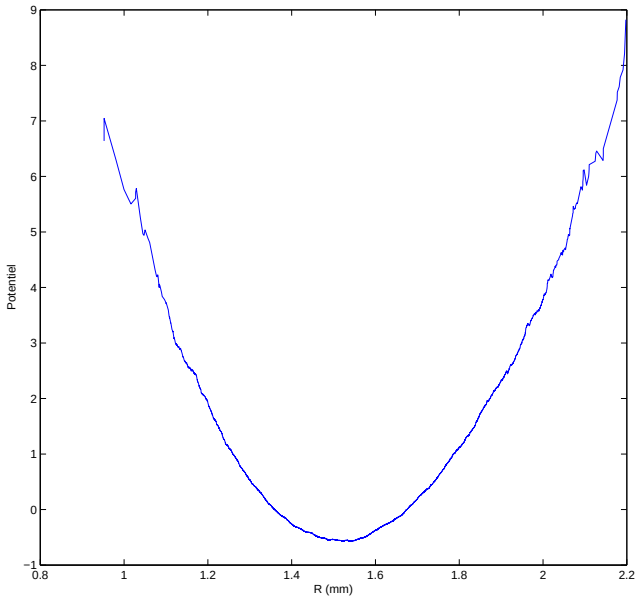

FIGURE 25 – Le potentiel que l'on obtient sans tenir compte de l'interaction bille-bord

III.2.4 Résultats

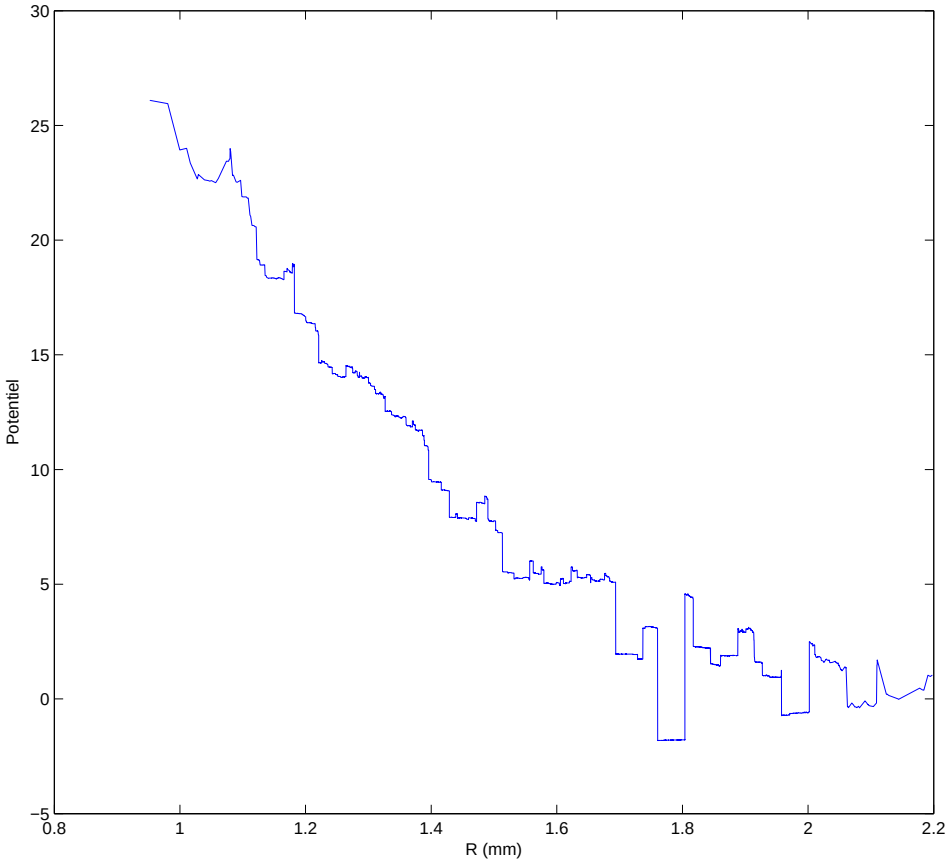

FIGURE 26 – Potentiel d'interaction entre deux billes

L'évolution est conforme à nos attentes : le potentiel est décroissant et tend vers une constante (nulle par choix de normalisation des probabilités).

Essayons de procéder à une régression. Notons r la distance entre les billes. Nous avons décidé de tester 3 modèles différents :

- Potentiel coulombien en $\frac{1}{r}$. En effet un premier modèle consiste à considérer que les billes acquièrent une charge que l'on décrète indépendante de la proximité à l'autre bille. Un tel modèle donne un potentiel coulombien.
- Potentiel coulombien écranté en $\frac{\exp(-\frac{r}{\lambda})}{r}$, où λ est un paramètre. Il n'y a aucune raison à priori de choisir un tel modèle, si ce n'est que la décroissance est plus rapide que pour le potentiel coulombien.
- Potentiel en $\epsilon_0 K_0(\frac{r}{\lambda})$, où K_0 est la fonction de Bessel modifiée de seconde espèce et ϵ_0 et λ des paramètres. Un tel potentiel a été trouvé par une méthode numérique par Gwennou Coupier dans sa thèse [1]).

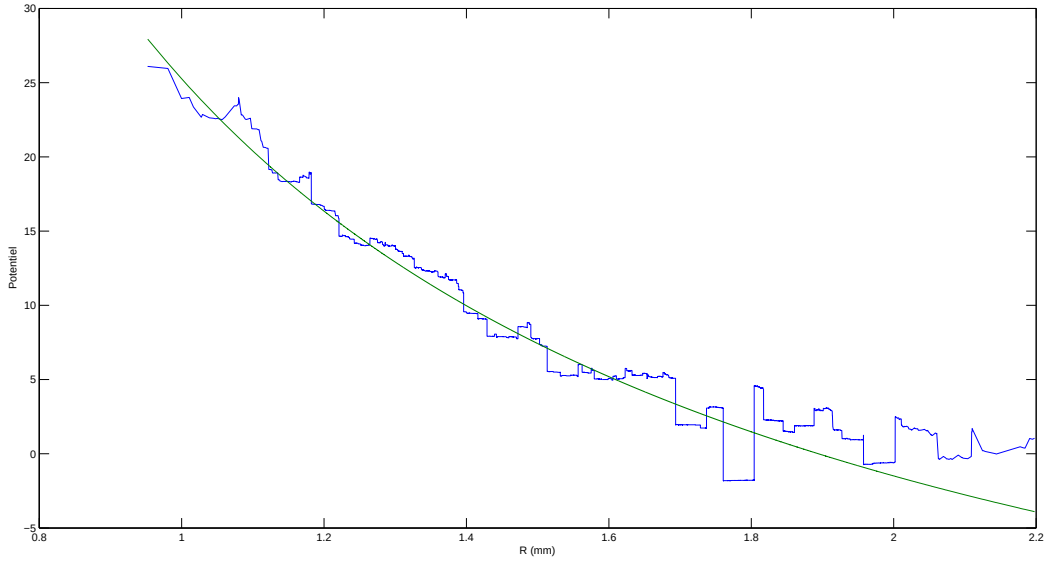

FIGURE 27 – Potentiel d'interaction entre deux billes modélisé par un potentiel coulombien : $V = A\frac{1}{r} + V_0$ avec $A = 53,5\ mm^{-1}$ et $V_0 = -28,2$ (les unités de potentiels sont arbitraires)

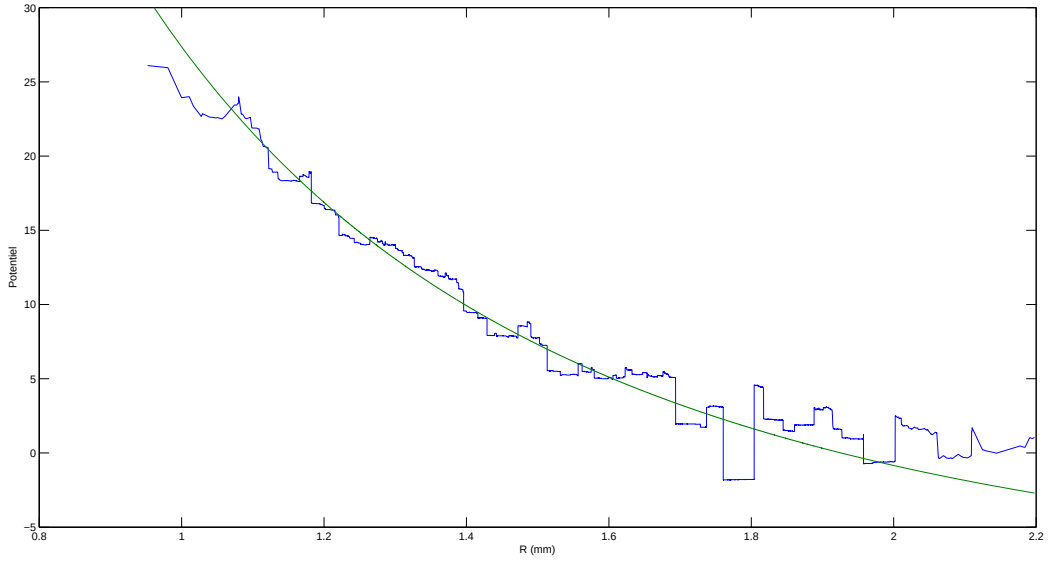

FIGURE 28 – Potentiel d'interaction entre deux billes modélisé par un potentiel coulombien écranté : $V = A\frac{\exp(-\frac{r}{\lambda})}{r} + V_0$ avec $A = 79,2\ mm^{-1}$, $V_0 = 9,3$ et $\lambda = 1,30\ mm$

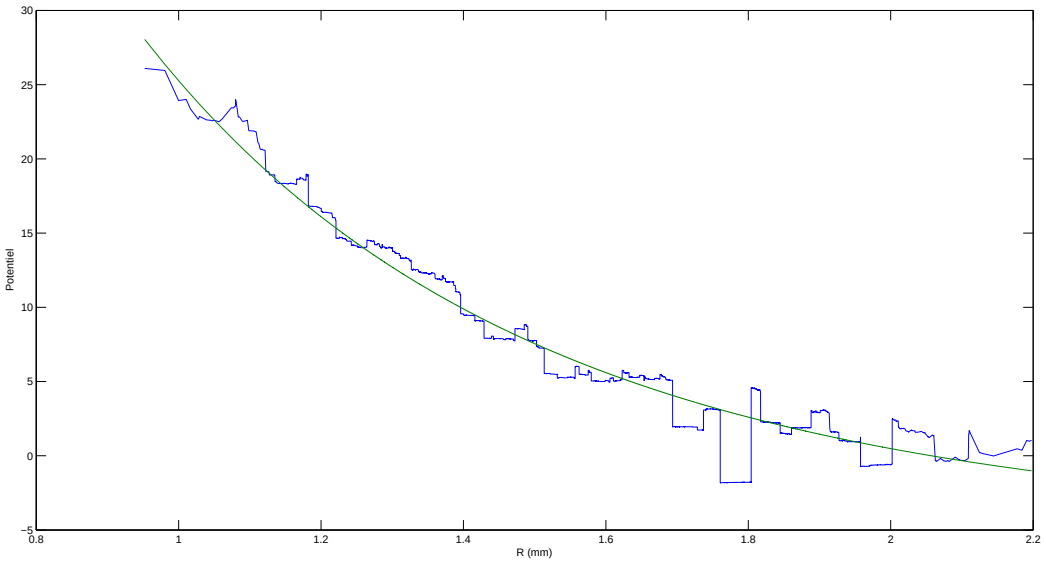

FIGURE 29 – Potentiel d'interaction entre deux billes modélisé par la fonction K_0 de Bessel : $V = \epsilon_0 K_0(\frac{r}{\lambda}) + V_0$ avec $V_0 = -5,0$; $\epsilon_0 = 123,7$ et $\lambda = 0,72\ mm$

III.3 Conclusion et perspectives

Dans cette partie, nous avons obtenu un tracé direct du potentiel d'interaction entre deux billes. La fonction K_0 de Bessel est le meilleur des 3 modèles que nous avons testé. Rappelons que ce modèle a été obtenu par des méthodes de calcul numérique par Gwennou Coupier [1].

Le résultat obtenu est tout à fait cohérent avec ce que l'on pouvait attendre. Ceci dit, il pourrait certainement être amélioré. Si le potentiel a été relativement bien caractérisé pour des distances entre les billes relativement faibles, le comportement à longue portée est plutôt mal décrit. Mais la taille du trou et la valeur du potentiel ont été imposés par les expériences déjà réalisées (nous n'avons malheureusement pas réussi à nous libérer de la contingence temporelle pour réaliser de nouvelles expériences). Le résultat est que le potentiel imposé par les bords du trou se comportait comme un puits harmonique alors que pour explorer les grandes distances, il aurait été préférable d'avoir un potentiel extérieur quasiment constant. Si l'expérience était à recommencer, nous la réaliserions dans le plus grand trou possible.

IV Bilan

Au terme de ce rapport, nous avons un petit aperçu de ce qu'il est possible de faire en terme de modélisation cristalline bidimensionnelle. Le modèle que nous avons étudié a d'ailleurs été étendu à de plus grandes surfaces et à un plus grand nombre de billes pour étudier des limites thermodynamiques, et étudier des comportements à plus grandes échelles.

Cependant nous avons :

- D'abord pu montrer que le modèle réduit expérimental modélisait bien un bain thermique : c'est à dire que l'on pouvait contrôler la vitesse quadratique moyenne des billes grâce à notre Agitation « thermique ». De plus, nos billes respectaient la statistique de Maxwell-Boltzmann.
- Ensuite nous avons étudié un petit système de cristal : six billes dans un pentagone. Dans le cadre de ce système nous avons pu introduire des notions de grandeurs énergétiques avec succès :
  - une énergie caractérisant la différence entre les états H et P.
  - une énergie caractérisant le temps de vie d'un état H.
- Finalement nous avons pu étudier l'influence de l'ajout du potentiel de 1000 V sur une bille, et l'utilisant à profit, nous avons vu ce que le dispositif engendrait comme type de potentiel d'interaction entre deux billes : dans une bonne approximation, on peut dire qu'il s'agit d'un potentiel écranté, ce que l'on utilise dans de nombreux modèles.

Ce fut un grand et beau projet, qui nous a beaucoup formé à différents langages informatiques et qui nous a appris beaucoup en termes de réflexions scientifiques thermodynamiques et probabilistes.

V Annexes

Code 1 : Écriture dans un seul fichier de la trajectoire des billes - suppression des erreurs commises par IDL

```
    % CHARGEMENT DES POSITIONS DES BILLES

function pos=chargepos(nbi,h)
j=0 ;
for i=1 :length(h)
    a=load(h{i},'-ascii') ;
    if (size(a,1)==nbi)
        j=j+1 ;
        pos(j).x=a( :,1) ;
        pos(j).y=a( :,2) ;
    elseif (j>0 && size(a,1)> nbi)
        bonnecase=[] ;
        for l=1 :nbi
            ecartmin=((pos(j).x(l) - a(1,1))^2 + (pos(j).y(l) - a(1,2))^2) ;
            for k=2 :size(a,1)
                bonnecase(l)=1 ;
                ecartk=((pos(j).x(l) - a(k,1))^2 + (pos(j).y(l) - a(k,2))^2) ;
                if ecartk < ecartmin
                    ecartmin=ecartk ;
                    bonnecase(l)=k ;
                end
            end
        end
        j=j+1 ;
        bonnecase=sort(bonnecase) ;
        for m=1 :nbi
            b(m,1)=a(bonnecase(m),1) ;
            b(m,2)=a(bonnecase(m),2) ;
        end
        pos(j).x=b( :,1) ;
        pos(j).y=b( :,2) ;
    elseif j==0
        j=j+1 ;
        pos(j).x= 300 ;
        pos(j).y= 300 ;
    end

end
end
```

Code 2 : Régression par fonction caractéristique d'une densité gaussienne de probabilité

```
    %Elie Gouzien et Sophie Marbach présentent Fitcargauss.
%Fitcargauss fits a integrated density probability to data and gives average and dispersion (as well as a normalized amplitude that shows sensitivity to statistical noise)
%Y=1/2+(1/2)*erf((X-M)/(Var*sqrt(2)))

function [FitPlot] = Fitcargauss(X1,Y1)

m = 4/3*(pi)*(0.0008)^3*7960 ; %masse de la bille
sigmav = 0.001 ; % ecart type sur l'écart entre vitesses obtenant en fitcargaussbis avec correlation de l'ordre de 10^{-5}

% Call fminsearch with a random starting point.
start_point = [0.2,0.3] ;
model = @cargauss ;
estimates = fminsearch(model, start_point) ;

assignin('base','Parameters',estimates) ;


function [sse, FittedCurve] = cargauss(params)
Moy = params(1) ;
Cartyp = params(2) ;
FittedCurve = (1/2+(1/2)*erf((X1-Moy)*0.001/(Cartyp*sqrt(2)))) ;
ErrorVector = FittedCurve - Y1 ;
sse = 0 ;
for i=1 :(length(ErrorVector) - 1)
    sse = sse + abs(((ErrorVector(i))^2)*(X1(i+1) - X1(i))) ;
end
kT = Cartyp^2*m ;
end

% cargauss accepts curve parameters as inputs, and outputs sse, the sum of squares error for modeldata-ydata, and the FittedCurve. FMINSEARCH only needs sse, but we want to plot the FittedCurve at the end.

pname=[inputname(2) ' vs ' inputname(1) ',Fit'] ;
% Create figure with handle
FitPlot=figure('PaperSize',[20.98 29.68],'Name',pname,'Color',[1 1 1]) ;

% Create plot
plot(X1,Y1,'x') ;
hold on
[sse,Fit]=cargauss(estimates) ;
plot(X1,Fit,'r') ;

% Create xlabel
xlabel(inputname(1),'FontSize',12) ;

% Create ylabel
```

```
ylabel(inputname(2),'FontSize',12);

%Display fit function and parameters
leg(1, :)= {['Correlation=', num2str(sse/sigmav)]};
leg(2, :)= {'[M,Cartyp]=' num2str(estimates)]};
leg(3, :)= {['k_B.T =' num2str(kT)]};

text(0.2,0.3,leg,'Units','normalized','HorizontalAlignment','right');

end
```

Code 3 : Régression par fonction caractéristique d'une densité exponentielle de probabilité

```
    %Permet de contrôler la répartition de Maxwell Boltzmann pour une bille sur un plan incliné

%Fitline fits a line to data and gives characteristics (as well as a normalized amplitude that shows sensitivity to statistical noise)
%Y=A*X+B

function [FitPlot] = Fitcarexp(X1,Y1)

m = 4/3*(pi)*(0.0008)^3*7960 ; %masse de la bille
g = 9.81 ; %pesanteur
alpha = 0.19/50 ; %angle d'inclinaison
sDeltax = 0.0127 ; %écart type sur l'écart des positions


%Optimisation sur le log pour obtenir des paramètres qui font converger la
%régression
start_point = [1,1] ;
model = @expp ;
estimates = fminsearch(model, start_point) ;

%régression elle même
start_point2 = [estimates(1),estimates(2)] ;
model2 = @expr ;
estimates2 = fminsearch(model2, start_point2) ;

assignin('base','Parameters',estimates) ;

function [sse, FittedCurve] = expp(params)
kT = params(1) ;
origine = params(2) ;
FittedCurve = -m*g*alpha*(X1-origine)*0.001/kT ;
ErrorVector = FittedCurve - log(Y1) ;
sse = 0 ;
for i=1 :(length(ErrorVector) - 1)
    sse = sse + abs(((ErrorVector(i))^2)*(X1(i+1) - X1(i))) ;
end
end

function [sse, FittedCurve] = expr(params)
kT = params(1) ;
origine = params(2) ;
FittedCurve = exp(-m*g*alpha*(X1-origine)*0.001/kT) ;
ErrorVector = FittedCurve - Y1 ;
sse = 0 ;
for i=1 :(length(ErrorVector) - 1)
    sse = sse + abs(((ErrorVector(i))^2)*(X1(i+1) - X1(i))) ;
end
end
```

```
% exp accepts curve parameters as inputs, and outputs sse,
% the sum of squares error for modeldata-ydata,
% and the FittedCurve. FMINSEARCH only needs sse, but we want
% to plot the FittedCurve at the end.

pname=[inputname(2) ' vs ' inputname(1) ',Fit'];
% Create figure with handle
FitPlot=figure('PaperSize',[20.98 29.68],'Name',pname,...
'Color',[1 1 1]);

% Create plot
plot(X1,Y1,'x');
hold on
[sse,Fit]=expr(estimates2);
plot(X1,Fit,'r');

% Create xlabel
xlabel(inputname(1),'FontSize',12);

% Create ylabel
ylabel(inputname(2),'FontSize',12);

%Display fit function and parameters
leg(1, :)= {['Corrélation=',num2str(sse/sDeltax)]};
leg(2, :)= {['kT =' num2str(estimates2(1)) ' J']};
leg(3, :)= {['Origine =' num2str(estimates2(2)) ' mm']};

text(0.5,0.1,leg,'Units','normalized','HorizontalAlignment','right');

end
```

Code 4 : Détermination de la durée de vie d'un état H

```
    %probabilité de vie d'un hexagone ou d'un pentagone

function [P] = TempsdevieH(Configuration)

% DEFINITION DES PARAMETRES

t=0.04 ; %temps entre images

%calcul des temps et stockage
H = 0 ; %booléen qui indique si on est en configuration H ou pas.
Tempsencours = 0 ;
j = 0 ;
for i = 1 :length(Configuration)
    if (strcmp(Configuration(i),'H')==1)&&(H ==0)
        H = 1 ;
        j = j+1 ;
    elseif (strcmp(Configuration(i),'H')==1)&&(H==1)
        Tempsencours = Tempsencours + 0.04 ;
    elseif (strcmp(Configuration(i),'H')==0) && (H==1)
        Tempsencours = Tempsencours + 0.04 ;
        Temps(j) = Tempsencours ;
        Tempsencours = 0 ;
        H=0 ;
    end
end


% écriture de la fonction caractéristique
Temps = sort(Temps) ;
N = length(Temps) ;
t=Temps(1) ;
k=1 ;
for j=1 :(N)
    if (t == Temps(j)) ;
    elseif (Temps(j) == t +0.04)
        P(k,1)=Temps(j-1) ;
        P(k,2)=1-(j-1)/N ;
        t=t+0.04 ;
        k = k +1 ;
    else
        P(k,1)=Temps(j-1) ;
        P(k,2)=1-(j-1)/N ;
        t = Temps(j) ;
        k = k+1 ;
    end
end

end
```

Code 5 : Première tentative de différenciation automatique entre H et P - distances entre les billes

```
    %"PouN" s'intéresse à la position des billes et à la distance aux autres billes en moyenne.
Ce programme n'est pas assez sensible, clairement.

function [Nombres,Configuration,P,H]= PouH()

%
%Chargement des positions : partie supprimée ici
%

% Références ! Attention à remplir par l'observateur...à partir de jolis
% états

for i = 28
    pos1(i,3)= 1/5*(sqrt((pos2(i,1)-pos1(i,1))^2 + (pos2(i,2)-pos1(i,2))^2) +sqrt((pos3(i,1)-pos1(i,1))^2
+ (pos3(i,2)-pos1(i,2))^2) +sqrt((pos4(i,1)-pos1(i,1))^2 + (pos4(i,2)-pos1(i,2))^2) +sqrt((pos5(i,1)-
pos1(i,1))^2 + (pos5(i,2)-pos1(i,2))^2) +sqrt((pos6(i,1)-pos1(i,1))^2 + (pos6(i,2)-pos1(i,2))^2)) ;
    pos2(i,3)= 1/5*(sqrt((pos2(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos4(i,1))^2 + (pos4(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos6(i,1))^2 + (pos6(i,2)-pos2(i,2))^2)) ;
    pos3(i,3)= 1/5*(sqrt((pos2(i,1)-pos3(i,1))^2 + (pos2(i,2)-pos3(i,2))^2)+sqrt((pos1(i,1)-pos3(i,1))^2
+ (pos1(i,2)-pos3(i,2))^2)+sqrt((pos4(i,1)-pos3(i,1))^2 + (pos4(i,2)-pos3(i,2))^2)+sqrt((pos5(i,1)-
pos3(i,1))^2 + (pos5(i,2)-pos3(i,2))^2)+sqrt((pos6(i,1)-pos3(i,1))^2 + (pos6(i,2)-pos3(i,2))^2)) ;
    pos4(i,3)= 1/5*(sqrt((pos4(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos2(i,1))^2 + (pos2(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos6(i,1))^2 + (pos6(i,2)-pos4(i,2))^2)) ;
    pos5(i,3)= 1/5*(sqrt((pos2(i,1)-pos5(i,1))^2 + (pos2(i,2)-pos5(i,2))^2)+sqrt((pos3(i,1)-pos5(i,1))^2
+ (pos3(i,2)-pos5(i,2))^2)+sqrt((pos4(i,1)-pos5(i,1))^2 + (pos4(i,2)-pos5(i,2))^2)+sqrt((pos1(i,1)-
pos5(i,1))^2 + (pos1(i,2)-pos5(i,2))^2)+sqrt((pos6(i,1)-pos5(i,1))^2 + (pos6(i,2)-pos5(i,2))^2)) ;
    pos6(i,3)= 1/5*(sqrt((pos6(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos4(i,1))^2 + (pos4(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos2(i,1))^2 + (pos2(i,2)-pos6(i,2))^2)) ;

    P = max([pos1(i,3),pos2(i,3),pos3(i,3),pos4(i,3),pos5(i,3),pos6(i,3)]) - min([pos1(i,3),pos2(i,3),pos3(i,3),pos4
end



for i = 158
    pos1(i,3)= 1/5*(sqrt((pos2(i,1)-pos1(i,1))^2 + (pos2(i,2)-pos1(i,2))^2)+sqrt((pos3(i,1)-pos1(i,1))^2
+ (pos3(i,2)-pos1(i,2))^2)+sqrt((pos4(i,1)-pos1(i,1))^2 + (pos4(i,2)-pos1(i,2))^2)+sqrt((pos5(i,1)-
pos1(i,1))^2 + (pos5(i,2)-pos1(i,2))^2)+sqrt((pos6(i,1)-pos1(i,1))^2 + (pos6(i,2)-pos1(i,2))^2)) ;
    pos2(i,3)= 1/5*(sqrt((pos2(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos4(i,1))^2 + (pos4(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos6(i,1))^2 + (pos6(i,2)-pos2(i,2))^2)) ;
    pos3(i,3)= 1/5*(sqrt((pos2(i,1)-pos3(i,1))^2 + (pos2(i,2)-pos3(i,2))^2)+sqrt((pos1(i,1)-pos3(i,1))^2
+ (pos1(i,2)-pos3(i,2))^2)+sqrt((pos4(i,1)-pos3(i,1))^2 + (pos4(i,2)-pos3(i,2))^2)+sqrt((pos5(i,1)-
pos3(i,1))^2 + (pos5(i,2)-pos3(i,2))^2)+sqrt((pos6(i,1)-pos3(i,1))^2 + (pos6(i,2)-pos3(i,2))^2)) ;
    pos4(i,3)= 1/5*(sqrt((pos4(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos2(i,1))^2 + (pos2(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-
```

```
pos5(i,1))^2 + (pos5(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos6(i,1))^2 + (pos6(i,2)-pos4(i,2))^2)) ;
    pos5(i,3)= 1/5*(sqrt((pos2(i,1)-pos5(i,1))^2 + (pos2(i,2)-pos5(i,2))^2)+sqrt((pos3(i,1)-pos5(i,1))^2
+ (pos3(i,2)-pos5(i,2))^2)+sqrt((pos4(i,1)-pos5(i,1))^2 + (pos4(i,2)-pos5(i,2))^2)+sqrt((pos1(i,1)-
pos5(i,1))^2 + (pos1(i,2)-pos5(i,2))^2)+sqrt((pos6(i,1)-pos5(i,1))^2 + (pos6(i,2)-pos5(i,2))^2)) ;
    pos6(i,3)= 1/5*(sqrt((pos6(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos4(i,1))^2 + (pos4(i,2)-pos6(i,2))^2) +sqrt((pos6(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos6(i,2))^2) +sqrt((pos6(i,1)-pos2(i,1))^2 + (pos2(i,2)-pos6(i,2))^2)) ;

    H = max([pos1(i,3),pos2(i,3),pos3(i,3),pos4(i,3),pos5(i,3),pos6(i,3)]) - min([pos1(i,3),pos2(i,3),pos3(i,3),pos4
end


% Calcul des sommes des distances moyennes aux autres billes

Configuration={} ;
NombreP=0 ;
NombreH=0 ;
NombreO=0 ;

for i=1 :length(pos1)
    pos1(i,3)= 1/5*(sqrt((pos2(i,1)-pos1(i,1))^2 + (pos2(i,2)-pos1(i,2))^2)+sqrt((pos3(i,1)-pos1(i,1))^2
+ (pos3(i,2)-pos1(i,2))^2)+sqrt((pos4(i,1)-pos1(i,1))^2 + (pos4(i,2)-pos1(i,2))^2)+sqrt((pos5(i,1)-
pos1(i,1))^2 + (pos5(i,2)-pos1(i,2))^2)+sqrt((pos6(i,1)-pos1(i,1))^2 + (pos6(i,2)-pos1(i,2))^2)) ;
    pos2(i,3)= 1/5*(sqrt((pos2(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos4(i,1))^2 + (pos4(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos2(i,2))^2)+sqrt((pos2(i,1)-pos6(i,1))^2 + (pos6(i,2)-pos2(i,2))^2)) ;
    pos3(i,3)= 1/5*(sqrt((pos2(i,1)-pos3(i,1))^2 + (pos2(i,2)-pos3(i,2))^2)+sqrt((pos1(i,1)-pos3(i,1))^2
+ (pos1(i,2)-pos3(i,2))^2)+sqrt((pos4(i,1)-pos3(i,1))^2 + (pos4(i,2)-pos3(i,2))^2)+sqrt((pos5(i,1)-
pos3(i,1))^2 + (pos5(i,2)-pos3(i,2))^2)+sqrt((pos6(i,1)-pos3(i,1))^2 + (pos6(i,2)-pos3(i,2))^2)) ;
    pos4(i,3)= 1/5*(sqrt((pos4(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos2(i,1))^2 + (pos2(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos4(i,2))^2)+sqrt((pos4(i,1)-pos6(i,1))^2 + (pos6(i,2)-pos4(i,2))^2)) ;
    pos5(i,3)= 1/5*(sqrt((pos2(i,1)-pos5(i,1))^2 + (pos2(i,2)-pos5(i,2))^2)+sqrt((pos3(i,1)-pos5(i,1))^2
+ (pos3(i,2)-pos5(i,2))^2)+sqrt((pos4(i,1)-pos5(i,1))^2 + (pos4(i,2)-pos5(i,2))^2)+sqrt((pos1(i,1)-
pos5(i,1))^2 + (pos1(i,2)-pos5(i,2))^2)+sqrt((pos6(i,1)-pos5(i,1))^2 + (pos6(i,2)-pos5(i,2))^2)) ;
    pos6(i,3)= 1/5*(sqrt((pos6(i,1)-pos1(i,1))^2 + (pos1(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos3(i,1))^2
+ (pos3(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos4(i,1))^2 + (pos4(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-
pos5(i,1))^2 + (pos5(i,2)-pos6(i,2))^2)+sqrt((pos6(i,1)-pos2(i,1))^2 + (pos2(i,2)-pos6(i,2))^2)) ;

    Distmin = min([pos1(i,3),pos2(i,3),pos3(i,3),pos4(i,3),pos5(i,3),pos6(i,3)]) ;
    Distmax = max([pos1(i,3),pos2(i,3),pos3(i,3),pos4(i,3),pos5(i,3),pos6(i,3)]) ;
    Dist = (Distmax-Distmin)/Distmax ;


    if Dist < 6.4003
        Configuration{i} = Dist ;
        NombreP = NombreP + 1 ;
    elseif Dist > 6.5
        NombreH = NombreH +1 ;
        Configuration{i} = 'H' ;
    else NombreO = NombreO + 1 ;
```

```
        Configuration{i} = '0';

    end
end

Nombres=[NombreP,NombreH,NombreO];
Configuration=Configuration';

end
```

Code 6 : Deuxième tentative de différenciation automatique entre H et P - distance moyenne au barycentre

```
    %PouHbarycentre s'intéresse à la distance moyenne au barycentre des 6 billes.

function [Nombres,Configuration,H,P]= PouHbarycentre()

%
%Chargement des positions : partie supprimée ici
%

% Références ! Attention à remplir par l'observateur... à partir de jolis
% états

for i = 14
    bar(1) = 1/6*(pos1(i,1) + pos2(i,1) + pos3(i,1) + pos4(i,1) + pos5(i,1) + pos6(i,1)) ;
    bar(2) = 1/6*(pos1(i,2) + pos2(i,2) + pos3(i,2) + pos4(i,2) + pos5(i,2) + pos6(i,2)) ;
    distancebar1 = sqrt((pos1(i,1)-bar(1))^2+(pos1(i,2)-bar(2))^2) ;
    distancebar2 = sqrt((pos2(i,1)-bar(1))^2+(pos2(i,2)-bar(2))^2) ;
    distancebar3 = sqrt((pos3(i,1)-bar(1))^2+(pos3(i,2)-bar(2))^2) ;
    distancebar4 = sqrt((pos4(i,1)-bar(1))^2+(pos4(i,2)-bar(2))^2) ;
    distancebar5 = sqrt((pos5(i,1)-bar(1))^2+(pos5(i,2)-bar(2))^2) ;
    distancebar6 = sqrt((pos6(i,1)-bar(1))^2+(pos6(i,2)-bar(2))^2) ;
    distancebarmoy = 1/6*(distancebar1 + distancebar2 + distancebar3 + distancebar4 + dis-
tancebar5 + distancebar6) ;

    P= distancebarmoy ;
end

for i = 159
    bar(1) = 1/6*(pos1(i,1) + pos2(i,1) + pos3(i,1) + pos4(i,1) + pos5(i,1) + pos6(i,1)) ;
    bar(2) = 1/6*(pos1(i,2) + pos2(i,2) + pos3(i,2) + pos4(i,2) + pos5(i,2) + pos6(i,2)) ;
    distancebar1 = sqrt((pos1(i,1)-bar(1))^2+(pos1(i,2)-bar(2))^2) ;
    distancebar2 = sqrt((pos2(i,1)-bar(1))^2+(pos2(i,2)-bar(2))^2) ;
    distancebar3 = sqrt((pos3(i,1)-bar(1))^2+(pos3(i,2)-bar(2))^2) ;
    distancebar4 = sqrt((pos4(i,1)-bar(1))^2+(pos4(i,2)-bar(2))^2) ;
    distancebar5 = sqrt((pos5(i,1)-bar(1))^2+(pos5(i,2)-bar(2))^2) ;
    distancebar6 = sqrt((pos6(i,1)-bar(1))^2+(pos6(i,2)-bar(2))^2) ;
    distancebarmoy = 1/6*(distancebar1 + distancebar2 + distancebar3 + distancebar4 + dis-
tancebar5 + distancebar6) ;

    H= distancebarmoy ;
end


% Calcul des sommes des distances moyennes au barycentre

Configuration={} ;
NombreP=0 ;
NombreH=0 ;
NombreO=0 ;
bar=[] ;
```

```
for i=1 :length(pos1)
    bar(1) = 1/6*(pos1(i,1) + pos2(i,1) + pos3(i,1) + pos4(i,1) + pos5(i,1) + pos6(i,1)) ;
    bar(2) = 1/6*(pos1(i,2) + pos2(i,2) + pos3(i,2) + pos4(i,2) + pos5(i,2) + pos6(i,2)) ;
    distancebar1 = sqrt((pos1(i,1)-bar(1))^2+(pos1(i,2)-bar(2))^2) ;
    distancebar2 = sqrt((pos2(i,1)-bar(1))^2+(pos2(i,2)-bar(2))^2) ;
    distancebar3 = sqrt((pos3(i,1)-bar(1))^2+(pos3(i,2)-bar(2))^2) ;
    distancebar4 = sqrt((pos4(i,1)-bar(1))^2+(pos4(i,2)-bar(2))^2) ;
    distancebar5 = sqrt((pos5(i,1)-bar(1))^2+(pos5(i,2)-bar(2))^2) ;
    distancebar6 = sqrt((pos6(i,1)-bar(1))^2+(pos6(i,2)-bar(2))^2) ;
    distancebarmoy = 1/6*(distancebar1 + distancebar2 + distancebar3 + distancebar4 + dis-
tancebar5 + distancebar6)

    if (abs(H-distancebarmoy)/H > abs(P - distancebarmoy)/P && abs(P - distancebarmoy)/P
< 0.03 )
        Configuration{i} = 'P' ;
        NombreP = NombreP + 1 ;
    elseif abs(H-distancebarmoy)/H < 0.03
        NombreH = NombreH +1 ;
        Configuration{i} = 'H' ;
    else Configuration{i} = '0' ;
        NombreO = NombreO +1 ;
    end
end

Nombres=[NombreP,NombreH,NombreO] ;
Configuration=Configuration' ;

end
```

Code 7 : Solution retenue de différenciation automatique entre H et P - distance min et max au barycentre

```
    %PouHbarycentre s'intéresse à la distance au barycentre de la bille la plus
%proche du barycentre comparée à la plus grande.

function [Nombres,Configuration,NombreP]= PouHbarycentremin()

% DÉFINITION DES PARAMÉTRÉS

nbi=6 ; %nombre de billes
exp=600 ; %agitation
tension=1000 ; %tension
ns=1 ; %nombre de séries
nst=5 ; %première série
t=0.04 ; %temps entre images
% seuils optimisés :
% sH = 0,637
% sP = 0,165

%
%Chargement des données : supprimé ici
%

Configuration={} ;
NombreP=0 ;
NombreH=0 ;
NombreO=0 ;
bar=[] ;

for i=1 :length(pos1)
    bar(1) = 1/6*(pos1(i,1) + pos2(i,1) + pos3(i,1) + pos4(i,1) + pos5(i,1) + pos6(i,1)) ;
    bar(2) = 1/6*(pos1(i,2) + pos2(i,2) + pos3(i,2) + pos4(i,2) + pos5(i,2) + pos6(i,2)) ;
    distancebar1 = sqrt((pos1(i,1)-bar(1))^2+(pos1(i,2)-bar(2))^2) ;
    distancebar2 = sqrt((pos2(i,1)-bar(1))^2+(pos2(i,2)-bar(2))^2) ;
    distancebar3 = sqrt((pos3(i,1)-bar(1))^2+(pos3(i,2)-bar(2))^2) ;
    distancebar4 = sqrt((pos4(i,1)-bar(1))^2+(pos4(i,2)-bar(2))^2) ;
    distancebar5 = sqrt((pos5(i,1)-bar(1))^2+(pos5(i,2)-bar(2))^2) ;
    distancebar6 = sqrt((pos6(i,1)-bar(1))^2+(pos6(i,2)-bar(2))^2) ;
    distancebarmin = min([distancebar1,distancebar2,distancebar3,distancebar4,distancebar5,distancebar6]) ;
    distancebarmax = max([distancebar1,distancebar2,distancebar3,distancebar4,distancebar5,distancebar6]) ;

    R = distancebarmin / distancebarmax ;



    if R < 0.165
        Configuration{i} = 'P' ;
        NombreP = NombreP + 1 ;
    elseif R > 0.637
        NombreH = NombreH +1 ;
        Configuration{i} = 'H' ;
```

```
    else Configuration{i} = '0';
        NombreO = NombreO + 1;
    end
end

Nombres=NombreH/NombreP;
Configuration=Configuration';

end
```

Code 8 : Optimisation double des seuils

```
    %PouHbarycentremin s'intéresse à la distance au barycentre de la bille la
%plus proche du barycentre
%et à la plus éloignée. Il teste alors une variable entre 0 et 1. On va
%faire dans ce programme VARIER DEUX SEUILS. Qui vont de 0 à 1. Par échelles
%à régler au début du programme.
%Attention pour un nombre de seuils supérieur à 100, ça peut être long.
%Il faut compter une nuit pour 1000*1000.
%Un indice d'avancement a été intégré au programme.
%à l'origine le tracé était séparé pour ne pas avoir à attendre une nuit à
%chaque essai de paramètre graphique.

function [Seuil]= PouHbarycentreminseuildouble()
% DEFINITION DES PARAMETRES

nbi=6 ; %nombre de billes
exp=1000 ; %agitation
tension=1000 ; %tension
ns=5 ; %nombre de séries
nst=1 ; %première série
t=0.04 ; %temps entre images
echelle = 0.01 ; % 1/echelle = nombre de tests de seuils

% NOMENCLATURE DES DOSSIERS. A PRECISER !
root='D :\Projet_experimental_cristaux_2d\experiences\Clef_sophie_fin\Dossier1\12032012\' ;
dir1=sprintf('N%d-A%d-V%d',nbi,exp,tension) ;
dir=strcat(root,dir1) ;

%Concaténation des séries pour chaque bille
%
%Partie supprimée pour se concentrer sur le programme
%


% Calcul des sommes des distances moyennes au barycentre pour différentes
% valeurs de seuils et enregistrement des résultats.

N = 1/echelle ;

for j=1 :N
    for k=1 :j

        NombreP=0 ;
        NombreH=0 ;
        NombreO=0 ;
        bar=[] ;

        for i=1 :length(pos1)
            bar(1) = 1/6*(pos1(i,1) + pos2(i,1) + pos3(i,1) + pos4(i,1) + pos5(i,1) + pos6(i,1)) ;
            bar(2) = 1/6*(pos1(i,2) + pos2(i,2) + pos3(i,2) + pos4(i,2) + pos5(i,2) + pos6(i,2)) ;
            distancebar1 = sqrt((pos1(i,1)-bar(1))^2+(pos1(i,2)-bar(2))^2) ;
```

```
            distancebar2 = sqrt((pos2(i,1)-bar(1))^2+(pos2(i,2)-bar(2))^2);
            distancebar3 = sqrt((pos3(i,1)-bar(1))^2+(pos3(i,2)-bar(2))^2);
            distancebar4 = sqrt((pos4(i,1)-bar(1))^2+(pos4(i,2)-bar(2))^2);
            distancebar5 = sqrt((pos5(i,1)-bar(1))^2+(pos5(i,2)-bar(2))^2);
            distancebar6 = sqrt((pos6(i,1)-bar(1))^2+(pos6(i,2)-bar(2))^2);
            distancebarmin = min([distancebar1,distancebar2,distancebar3,distancebar4,distancebar5,distanceba
            distancebarmax = max([distancebar1,distancebar2,distancebar3,distancebar4,distancebar5,distanceb

            R = distancebarmin / distancebarmax;

            if R < k*echelle
                NombreP = NombreP + 1;
            elseif R > j*echelle
                NombreH = NombreH +1;
            else NombreO = NombreO + 1;
            end

        end


        Seuil(j,k,1) = NombreH;
        Seuil(j,k,2) = NombreP;
        Seuil(j,k,3) = NombreO;
    end
    avancement = j*(j+1)*echelle*echelle*100
end

%%%Plot
%a et b sont les limites de l'échelle de couleur. Il faudra donc les
%choisir pertinemment pour voir quelque chose dans la zone interessante.
%a=-1 et b=3 est pas mal pour tracer le plot HP
%plot3k est capable de les trouver automatiquement mais n'y arrive pas en
%raison de la divergence des grandeurs calculées.

%Remarquez le changement de variable : on s'intéresse désormais à la
%moyenne et à l'écart entre les seuils. Cela permet d'avoir des variables
%plus "découplées" (donc optimisation plus facile à voir).

a=-1;
b=3;

k = 1;
for j=1 :N
    for i=1 :j

        L(k,1) = (i+j)*echelle/2;
        L(k,2) = (j-i)*echelle;
        L(k,3) = log(Seuil(j,i,2))-log(Seuil(j,i,1));

        k = k+1;
```

```
    end
end

%exécution de la commande de tracé
plot3k(L,'ColorRange',[a,b], 'Labels',{'Optimisation des seuils : équilibre P-H','Moyenne des seuils','Écart entre les seuils','Energie (U.A.)',''}) ;

end
end
```

Code 9 : Détermination du potentiel bord-bille

```
    %"puitposcar" calcule la fonction caractéristique
%expérimentale pour la loi de probabilités radiale : Il calcule P(être à
%l'intérieur du cercle de rayon R) en fonction de R. Puis la dérive pour en
%tirer la densité de proba (utile pour corriger l'influence du potentiel
%sur ce qui se passe dans le piège) et l'énergie du piège.

%Il plotte : la proba, la densité radiale de probabilité et le profil
%énergétique du puit et son extrapolation.
%Rien n'a été spécialement normé : de toute
%façon tout revient à dire que 1) l'énergie est définie à une constante
%près 2) on a pas envie de connaitre kT : seul la forme des fonctions a un
%intérêt ici. Ceci dit, en pratique j'ai tout de même normé la fonction
%caractéristique donc également la densité. L'Energie reste elle non
%normée.

% La sortie 1 est la densité radiale de probabilité sous la forme d'une
% matrice ro contenant en ro( :,1) les abscisses de là où elle a été
% calculée et en ro( :,2) les ordonnées correspondantes. Elle sert aux
% vielles versions du potentiel à deux corps.

%La sortie 2 est l'énergie potentielle du puit. Son exponentielle est à
%injecter dans la correction pour éliminer l'effet du puit sur la
%statistique d'intéraction à deux corps. Elle est sous la forme d'un
%polynôme de degrès 2.

%type de dérivation retenue : fenêtre glissante de taille fixe sur l'axe
%des abscisses puis la dérivée est obtenue par taux d'accroissement entre
%les extrémités de la fenêtre. Le point est attribué au milieu de la
%fenêtre.

function [ro,Energie]= puitsposcar()

% DEFINITION DES PARAMETRES

nbi=1 ; %nombre de billes
exp=800 ; %agitation
tension=1000 ; %tension
ns=5 ; %nombre de séries ; afin de garder kT constant, on ne travail qu'avec une agitation donnée

nst=1 ; %première série
t=0.04 ; %temps entre images
resolution=20 ; %(rmax-rmin)/resolution est la taille de la fenêtre. Résolution correspond au
nombre de fenêtres necessaires au recouvrement de l'interval de définition de la fonction que l'on
dérive (ici la fonction caractéristique).

% NOMENCLATURE DES DOSSIERS. A PRECISER !
root='D :\Projet_experimental_cristaux_2d\experiences\Clef_sophie_fin\Dossier1\16032012\' ;
dir1=sprintf('N%d-A%d-V%d',nbi,exp,tension) ;
dir=strcat(root,dir1) ;
```

```
%Concaténation des séries
for i=nst :ns+nst-1
    h{i}=strcat(dir,sprintf('/traj_A%d_Serie%d_B1.txt',exp,i)) ;
end
pos=[] ;
for i=nst :length(h)
    a=load(h{i},'-ascii') ;
    p=length(pos) ;
    for j=1 :length(a)
        pos(j+p,1)=a(j,1)-272/10 ;
        pos(j+p,2)=a(j,2)-26/10 ;
    end

end

%Détermination du centre en calculant la position du barycentre.
bar =[] ;
bar(1) = 0 ;
bar(2) = 0 ;
N = length(pos) ;
for i = 1 :N
    bar(1) = bar(1)+ pos(i,1)/N ;
    bar(2) = bar(2)+ pos(i,2)/N ;
end

%Colonne des distances au barycentre (rayons)
for i=1 :N
    r(i)=sqrt((pos(i,1)-bar(1))^2+(pos(i,2)-bar(2))^2) ;
end

%création de la liste des probabilités (comprendre fonction caractéristique)
r = sort(r) ; %en fait il n'a pas l'air d'aimer sortrows
for i=1 :N
    car(i,1)=r(i) ;
    car(i,2)=i/N ;
end

%plot de la fonction caractéristique
figure('name','Caractéristique radiale') ;
plot(car( :,1),car( :,2)) ;

%calcul de la taille de la fenêtre
taille = (car(length(car( :,1)),1)-car(1,1))/resolution ;

%dérivation
i=1 ;
while car(i,1)+taille <= car(length(car( :,1)),1)
    j=i ;
    while car(j,1) < car(i,1)+taille
        j=j+1 ;
    end
```

```
    ro(i,1)=(car(i,1)+car(j,1))/2;
    ro(i,2)=(car(j,2)-car(i,2))/(car(j,1)-car(i,1));
    i=i+1;
end

%no comment
figure('name','Densité de probabilité');
plot(ro(:,1),ro(:,2));

%Et en bonus, tracé du profil énergétique du puit.
E(:,1)=ro(:,1);
E(:,2)=log(ro(:,1))-log(ro(:,2));
figure('name','Profil énergétique du puit (U.A.)');
hold on
plot(E(:,1), E(:,2));

%extrapolation du profil énergétique
Energie = polyfit(E(:,1),E(:,2),2);
plot(E(:,1), polyval(Energie,E(:,1)));

%La suite a été écrite pour voir si éliminer le terme en x était pertinant.
%copier collé délicat depuis carexp
%ceci est un fit en ax^2+b de l'énergie
start_point = [14,-4];
model = @binome;
estimates = fminsearch(model, start_point);
function [sse, FittedCurve] = binome(params)
a = params(1);
b = params(2);
FittedCurve = a*E(:,1).*E(:,1) + b;
ErrorVector = FittedCurve - E(:,2);
sse = 0;
for i=1:(length(ErrorVector))
    sse = sse + ErrorVector(i)^2;
end
end

Energiebis=Energie;
Energiebis(1)=estimates(1);
Energiebis(2)=0;
Energibis(3)=estimates(2);

plot(E(:,1), polyval(Energiebis,E(:,1)),'g');
title('En bleu : energie expérimentale et extrapolée en vert : extrapolation sans terme d'ordre 1')

end
```

Code 10 : Détermination du potentiel bille-bille

```
    %"Interactiondeuxbilles" trouve le potentiel d'interaction à partir
%de la densité de probabilité pour une bille dans le potentiel extérieur et
%des positions de deux billes.


%Il trace : Tout corrigé et naïf (sans prendre en compte la présence du
%potentiel extérieur) : les fonctions caractéristique, la densité radiale de
%probabilité (la dérivée de la caractéristique corrigée) et le profil
%énergétique d'interaction (et ses régressions).
%Rien n'est convenablement normé : de toute façon tout revient à dire que
%1) L'énergie est définie à une constante près
%2) On a pas envie de connaitre kT : seul la forme des fonctions a un
%intérêt ici. On travaille donc à un facteur de dillatation près.
%3) De toute façon dans un espace infini, on ne peut pas normer car à
%l'infini, l'énergie reste finie. On norme donc sur le compact dans lequel
%on a des points, même si en toute rigueur la probabilité d'avoir des
%points en dehors est non nulle.


% Les sorties sont l'énergie d'interaction et les paramètres de ces modélisations.

%Le type de dérivation utilisée est une dérivation par taux d'accroissement
%les bords d'une fenêtre glissante à dimension horizontale fixée. Même méthode que pour le puitderivee5.

%L'entrée est l'énergie du puits. Il est rentré sous la forme d'un polynôme
%de second degrés.

function [E,estimates,estimates2,estimates3]= Interactiondeuxbilles(Epuit)

% DEFINITION DES PARAMETRES

nbi=2 ; %nombre de billes
agit=800 ; %agitation
tension=1000 ; %tension
ns=10 ; %nombre de séries afin de garder kT constant, on ne travail qu'avec une agitation donnée

nst=1 ; %première série
t=0.04 ; %temps entre images
resolution=30 ; %30 %paramètre de lissage pour la dérivation (optimisé visuellement à 30).
%(rmax-rmin)/resolution est la taille de la fenêtre. Résolution correspond
%au nombre de fenêtres nécessaires au recouvrement de l'interval de
%définition de la fonction que l'on dérive (ici la fonction caractéristique).
debut=0.9 ; %sur nos expériences, 0,9 est bon % Les rayons trop faibles correspondent aux erreur sur le positionnement des points, il faut éliminer ces images
fin=10 ; %a partir de 2 : sans effet %A priori, pas de problème à la fin ; on se garde la possibilité d'intervenir
rayonmax=50 ; %distance centre-bille pour une des billes à partir de laquelle le point est supprimé. Attention à ne pas fausser la statistique
```

```
% NOMENCLATURE DES DOSSIERS. A PRÉCISER !
root='D :\Projet_experimental_cristaux_2d\experiences\Clef_sophie_fin\Dossier1\19032012\';
dir1=sprintf('N%d-A%d-V%d',nbi,agit,tension);
dir=strcat(root,dir1);

%Concaténation des séries pour chaque bille.
%bille 1
for i=nst :ns+nst-1
    h{i}=strcat(dir,sprintf('/traj_A%d_Serie%d_B1.txt',agit,i));
end
pos1=[];
for i=nst :length(h)
    a=load(h{i},'-ascii');
    p=length(pos1);
    for j=1 :length(a)
        pos1(j+p,1)=a(j,1)-272/10;
        pos1(j+p,2)=a(j,2)-26/10;
    end
end

%bille 2
for i=nst :ns+nst-1
    h{i}=strcat(dir,sprintf('/traj_A%d_Serie%d_B2.txt',agit,i));
end
pos2=[];
for i=nst :length(h)
    a=load(h{i},'-ascii');
    p=length(pos2);
    for j=1 :length(a)
        pos2(j+p,1)=a(j,1)-272/10;
        pos2(j+p,2)=a(j,2)-26/10;
    end
end


%Détermination du centre par calcul de la position du barycentre
bar =[];
bar(1) = 0;
bar(2) = 0;
N = length(pos1);
for i = 1 :N
    bar(1) = bar(1)+ pos1(i,1)/(2*N)+pos2(i,1)/(2*N);
    bar(2) = bar(2)+ pos1(i,2)/(2*N)+pos2(i,2)/(2*N);
end


%Calcul des grandeurs intéressantes
for i=1 :N
    distances(i,1)=sqrt((pos1(i,1)-pos2(i,1))^2+(pos1(i,2)-pos2(i,2))^2);
    distances(i,2)=sqrt((pos1(i,1)-bar(1))^2+(pos1(i,2)-bar(2))^2);
    distances(i,3)=sqrt((pos2(i,1)-bar(1))^2+(pos2(i,2)-bar(2))^2);
```

```
end



%%%%Calcul de la variable aléatoire qui se moyenne en "probabilité
%%%%microscopique corrigée". Elle est vouée à être sommée.
%trie de la liste
distances=sortrows(distances,1) ;

%Calcul de la fonction caractéristique
p=1 ;
while distances(p,1)<debut
    p=p+1 ;
end
car=[] ;
car(1,1)=distances(p,1) ;
car(1,2)=exp(polyval(Epuit,distances(p,2)))*exp(polyval(Epuit,distances(p,3))) ;
car(1,3)=1 ;
j=2 ;
for i=p :N
    if fin<distances(i,1) || rayonmax<distances(i,2) || rayonmax<distances(i,3)
        break
    end
    car(j,1)=distances(i,1) ;
    car(j,2)=car(j-1,2)+exp(polyval(Epuit,distances(i,2)))*exp(polyval(Epuit,distances(i,3))) ;
    car(j,3)=j ;
    j=j+1 ;
end


%normalisation (tout à fait licite).
for i=1 :length(car( :,1))
    car(i,2)=car(i,2)/car(length(car( :,1)),2) ;
    car(i,3)=car(i,3)/length(car( :,1)) ;
end

%plot de la fonction caractéristique naïve
figure('name','caractéristique naïve') ;
plot(car( :,1),car( :,3)) ;

%plot de la fonction caractéristique corrigée
figure('name','caractéristique corrigée') ;
plot(car( :,1),car( :,2)) ;



%%%début de la dérivation%%%%

%calcul de la taille de la fenêtre
taille = (car(length(car( :,1)),1)-car(1,1))/resolution ;
```

```
%dérivation
i=1;
while car(i,1)+taille <= car(length(car(:,1)),1)
    j=i;
    while car(j,1) < car(i,1)+taille
        j=j+1;
    end
    ro2(i,1)=(car(i,1)+car(j,1))/2;
    ro2(i,2)=(car(j,2)-car(i,2))/(car(j,1)-car(i,1));
    ro2(i,3)=(car(j,3)-car(i,3))/(car(j,1)-car(i,1));
    i=i+1;
end

%no comment
figure('name','Densité de probabilité');
plot(ro2(:,1),ro2(:,2));

figure('name','Densité de probabilité naive');
plot(ro2(:,1),ro2(:,3));

%Tracé du profil énergétique
E(:,1)=ro2(:,1);
E(:,2)=log(ro2(:,1))-log(ro2(:,2));
E(:,3)=log(ro2(:,1))-log(ro2(:,3));

figure('name','Profil énergétique d'interaction naive');
plot(E(:,1), E(:,3));



%%%%%modèles de potentiel%%%%

%Potentiel coulombien
%ceci est un fit en a/x de l'énergie
function [sse, FittedCurve] = coulomb(params)
a = params(1);
b = params(2);
%c = params(3);
c = 0;
FittedCurve = [];
sse = 0;
ErrorVector = [];
for i=1 :length(E(:,1))
    FittedCurve(i,1) = a/(E(i,1)-c) + b;
    ErrorVector(i,1) = FittedCurve(i,1) - E(i,2);
    sse = sse + ErrorVector(i)^2;
end
end

%valeurs obtenues à partir d'une optimisation
start_point = [53.506808104768155,28.24561116657246];
```

```
model = @coulomb;
estimates = fminsearch(model, start_point);
[sse, E( :,4)] = coulomb(estimates);

figure('name','modélisation en potentiel Coulombien');
plot( E( :,1), E( :,2),E( :,1), E( :,4));


%Potentiel écranté
%ceci est un fit en b+(a/(x))*exp(-(x)/d de l'énergie
function [sse, FittedCurve] = yukawa(params)
a = params(1);
b = params(2);
d = params(3);
%c = params(4);
c = 0;
FittedCurve = [];
sse = 0;
ErrorVector = [];
for i=1 :length(E( :,1))
    FittedCurve(i,1) = a*exp(-(E(i,1)-c)/d)/(E(i,1)-c) + b;
    ErrorVector(i,1) = FittedCurve(i,1) - E(i,2);
    sse = sse + ErrorVector(i)^2;
end
end

start_point = [79.21125735384078,9.348038509305717,1.300091066337536];
model2 = @yukawa;
estimates2 = fminsearch(model2, start_point);
[sse, E( :,5)] = yukawa(estimates2);

figure('name','Profil énergétique d'interaction ; modélisation yukawa');
plot( E( :,1), E( :,2),E( :,1), E( :,5));


%Potentiel en fonction de besselk0.
%ceci est un fit en E0*besselk(0,X/lambda) de l'énergie
function [sse, FittedCurve] = potentielbessel(params)
a = params(1);
b = params(2);
d = params(3);
%c = params(3);
c = 0;
FittedCurve = [];
sse = 0;
ErrorVector = [];
for i=1 :length(E( :,1))
    FittedCurve(i,1) = a*besselk(0,(E(i,1)-c)/d) + b;
    ErrorVector(i,1) = FittedCurve(i,1) - E(i,2);
    sse = sse + ErrorVector(i)^2;
end
```

```
end

start_point = [236.6223187135314,4.258405868745415,0.552455057007836];
model3 = @potentielbessel;
estimates3 = fminsearch(model3, start_point);
[sse, E( :,6)] = potentielbessel(estimates3);

figure('name','Profil énergétique d'interaction ; Modélisation Bessel');
plot( E( :,1), E( :,2),E( :,1), E( :,6));

end
```

Références

[1] Élasticité et ancrage dans des cristaux de Wigner macroscopiques : un système modèle pour l'étude du piégeage faible ;
Gwennou Coupier - Thèse de Physique du solide, Université Paris 6, 2006