DYNAMIQUE MOLECULAIRE

Le but de ce projet est de modéliser la dynamique d'un ensemble de particules formant un gaz. Sans perdre en généralité, on suppose qu'elles se déplacent dans un espace de dimension 2. Dans cette simulation, il est nécessaire de décrire les interactions entre particules. Ici, on suppose qu'elles sont de type cœur dur, autrement dit lorsque deux particules entrent en contact, la quantité totale de mouvement et l'énergie cinétique totale avant et après choc sont conservées.

Les deux objectifs principaux sont i) l'écriture des équations physiques du mouvement balistique des particules et des lois gouvernant les chocs, ii) l'écriture de l'algorithme de principe du programme de simulation avant son développement en C iii) la comparaison des distributions de vitesses obtenues avec celles attendues en physique statistique.

Hypothèses :

- Les particules sont disposées dans une boite de taille [-L, L]x[-L, L] avec L=1. On applique des conditions périodiques sur les bords de la boite, autrement dit toute particule sortant de la boite par un bord donné est réinjecté immédiatement par le bord opposé.

- Initialement, les vitesses en module de toutes les particules sont identiques et valent v0=0.1. Le vecteur associé est uniformément distribué sur un cercle de rayon v0.

- Deux particules se trouvant à une distance inférieure à ε=0.01*L sont en contact et subissent un choc élastique. Leur mouvement est modifié en conséquence.

Principe du programme :

- Après la phase d'initialisation
  - Tirage de la position initiale des N particules. On prendra N de l'ordre de 200. Pour tout i tirer une position x(i) $\in$ [-L, L] et y(i) $\in$ [-L, L]
  - Tirage de la vitesse des particules $v_x(i)=v0*\cos(\phi)$ et $v_y(i)=v0*\sin(\phi)$ avec $\phi=2\pi r$ où r est un nombre aléatoire r$\in$ [0, 1[
-
- Boucle infinie sur le nombre d'itération
  - Calcul du temps τmin instant de la première collision entre deux particules.et identification de la paire de particules entrant en collision.
  - Mise à jour du temps : t = t+τmin.
  - Evolution de la position des particules : elles effectuent un mouvement rectiligne pendant τmin.
  - Traitement du choc élastique de la paire précédemment identifiée.
  - Mise à jour de la distribution des particules selon leur vitesse selon Ox ou Oy.
- Fin de boucle sur le nombre d'itération.

Travail préliminaire :

- Définir une structure de données regroupant la position et la vitesse d'une particule.
- Quelle est la structure de données la mieux adaptée pour décrire une distribution de particules du point de vitesse d'exécution et temps d'accès aux données d'une particule ?

- Ecrire un module algorithmique déterminant à partir d'une distribution de particules en position et en vitesse le 1er instant de collision τmin entre deux particules. Il retournera le temps τmin et identifiera les deux particules qui entrent en collision.
- Ecrire un algorithme de mise à jour de la position des particules.
- Ecrire un algorithme de traitement des chocs élastiques.

Installation de la librairie de calcul scientifique gsl :

Installation de la librairie mathématique gsl-latest :
Copier gsl-latest.tar.gz dans votre Home Directory
Décompresser l'archive localement dans votre Home Directory en tapant
tar –xzf gsl-latest.tar.gz
Placer vous dans le répertoire gsl-latest puis taper
./configure --prefix=$HOME/ **extralib**
make

Lire documentation sur internet concernant le tirage de nombres aléatoires et l'interprétation de données sous forme d'histogramme. Tester le comportement de la librairie en créant de petits programmes tests comme celui donné ci-après :

```
#include <stdio.h>
#include <math.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>
#include <gsl/gsl_histogram.h>

using namespace std;

gsl_rng *r=NULL;                    /* déclaration globale du generateur */
unsigned long int seed = 1564431;   /* nombre source du generateur */

main ()
{
int i;
int Ntirages=10000;                 /* nb de tirages */
int Nintervalles=50;                /* nb d'intervalles de l'histogramme */

const gsl_rng_type * T=NULL;        /* type de générateur */
gsl_histogram * h=NULL;             /* histogramme */

gsl_rng_env_setup();

T = gsl_rng_default;
r = gsl_rng_alloc (T);
printf("generator type : %s\n", gsl_rng_name (r));

/* initialisation du generateur avec le nombre source */
```

```
gsl_rng_set(r, seed);


h = gsl_histogram_alloc (Nintervalles);

/* domaine de stockage [-10, 10] */
gsl_histogram_set_ranges_uniform (h, -10, 10);

for (i=0; i<Ntirages; i++)
    {
/*  tirage d'un nb aleatoire distrib uniforme */
/*  double x=gsl_rng_uniform (r);  */

/* tirage d'un nb aleatoire distrib gaussienne */
    double x=gsl_ran_gaussian (r, 1.);
    gsl_histogram_increment(h, x);
    }

for (i=0; i<Nintervalles; i++)
    {
    printf("%d\t%f\t%f\n", i, 0.5*(h->range[i]+h->range[i+1]), h->bin[i]);
    }

gsl_histogram_free (h);
getchar();
}
```

Développement du simulateur en C:

- Lire Algo_Dynamique_Moleculaire.pdf pour comprendre l'algorithme de principe.
  Si nécessaire, installer octave sous windows ou linux et exécuter le programme Maxwell_main.m qui est un simulateur écrit en Matlab/Octave.

- On vous propose de stocker les coordonnées ainsi que les vitesses d'une particule dans une structure Particule. Les informations nécessaires pour décrire le gaz de particules sont stockés dans un tableau de structures Particule.

- Pour vous aider, on vous impose de plus d'utiliser les prototypes suivants :

void InitDistrib(int N, Particule *part);
pour initialiser le gaz.
void PredictCollision(int N, Particule *part, int *npart1, int *npart2, double *taumin);
pour déterminer le temps du premier choc.
void EvolutionTempo(int N, double tau, Particule *part);
pour faire avancer les particules entre 2 chocs.
void TraitCollision(int N, int npart1, int npart2, Particule *part);
pour traiter les collitions.

- Phase de développement : traduire votre algorithmique en C en définissant et commentant bien les interfaces des fonctions programmées. Comme pendant le premier semestre, un développement incrémental, module par module est souhaitable.

- Phase de tests : la première version de votre programme est purement graphique. Elle montre l'évolution en direct des particules à l'écran.

- Phase d'exploitation : notre but est aussi de faire de la physique statistique en traçant la distribution de vitesses des particules. Calculer toutes les 100 itérations l'histogramme associé à la distribution de vitesse en x ou en y. Stocker-le régulièrement dans un fichier et tracer-le grâce au logiciel gnuplot.

- Phase d'extension : à vous de proposer des améliorations voire de nouvelles études physiques.