Computational Physics - Project 5

Johannes Scheller (candidate no. 71), Vincent Noculak (candidate no. 22)
Lukas Powalla (candidate no. 67), Richard Asbah (candidate no. 50)

December 11, 2015

Contents

1 Execution

in order to analysis our data we need to find the potential energy and the kinetic energy at the time t. Down here is the KinpotEnergy void function which calculate the kinetic energy and potential energy for each particle. When ever this function i called in RK4 or verlet method it calculate this list for a time t.

```
kineticEnergy <vector> //where we store each particle kinetic energy
potentialEnergy <vector> //where we store each particle potential energy

theTotalEnergy // where we store the total energy for all particles
total_kin // where we store the total kinetic energy only for the particles in system
total_pot // where we store the total potential energy only for the particles in system
numplanetsInSystem // the number of particles still in the system
ergodic //the value of ergodic ratio
```

```
void solarsystem::kinPotEnergy(){

    kineticEnergy = new double[ this->numplanets];
    total_kin = 0.0;
    total_pot = 0.0;

    double potenial = 0.0;
    double totalKinetic = 0.0;
    double totalPotenial = 0.0;
    double velocitySquared= 0.0;

    //kineticEnergy = 1/2 * m * v^2
    for (int i = 0; i < this->numplanets;i++){
        velocitySquared = A(3*i,1)*A(3*i,1)+A(3*i+1,1)*A(3*i+1,1)+A(3*i+2,1)*A(3*i+2,1);
        kineticEnergy[i] = 0.5*planets[i].m*velocitySquared;

        totalKinetic += kineticEnergy[i];
    }

    //potineal energy U = -G Mm/r-----------------
    potentialEnergy = new double[ this->numplanets];
    double r;
    Mat<double> p = Mat<double>(this->numplanets,this->numplanets,fill::zeros); //matrix p will includ

    for (int i = 0;i < this->numplanets; i++ ){
        for (int j = i+1; j < this->numplanets; j++){
            r = (A(3*i,0)- A(3*j,0))*(A(3*i,0)- A(3*j,0)) + (A(3*i+1,0)- A(3*j+1,0))*(A(3*i+1,0)- A(3*j
            r = sqrt(r);
            p(i,j)=-planets[i].m*planets[j].m*G/r;
            p(j,i)=p(i,j);
            totalPotenial += p(j,i);
        }
    }
    //calcuting the potenial energy for each planet from the marix P
    for(int i = 0; i < this->numplanets; i++){
        for(int j = 0; j < this->numplanets; j++){
            potenial += p(i,j);
        }
        potentialEnergy[i] = potenial;
        potenial = 0;
    }
    theTotalEnergy = totalKinetic+totalPotenial;

    //Virial analysis
     numplanetsInSystem = 0;
    //clac the energy of the bound system ! virial !!!
    for (int i = 0;i < this->numplanets; i++ ){
        if (( kineticEnergy[i]+potentialEnergy[i])<0.0){
            numplanetsInSystem += 1;
            total_kin += kineticEnergy[i];
            total_pot += potentialEnergy[i];
        }

    }
    total_pot = total_pot/2;
    ergodic = total_pot/total_kin;
}
```

in the first part pf the function we calculate the kinetic energy for each particle as $1\frac{1}{2}mv(t)^2$
in the second part we calculate all the possible potential energy between different particles in a matrix p.

$$p = \begin{bmatrix} 0 & \frac{-Gm_1m_0}{r} & \frac{-Gm_2m_0}{r} & ... & \frac{-Gm_nm_0}{r} \\ \frac{-Gm_0m_1}{r} & 0 & \frac{-Gm_2m_1}{r} & ... & \frac{-Gm_nm_1}{r} \\ ... & ... & ... & ... & ... \\ \frac{-Gm_0m_n}{r} & \frac{-Gm_1m_n}{r} & \frac{-Gm_2m_n}{r} & ... & 0 \end{bmatrix} \tag{1}$$

then we make new for loop, potential energy for a particle n is the sum over p nrow, and the last step in this function is to calculate the total potential and kinetic of the system by summing over all the bounded particle with the if test "only the particles with minus total energy".