TRAJECTOIRE DES PLANÈTES

L'informatique étant un puissant moyen de calcul et de visualisation, il est facile avec quelques connaissances de programmation en BASIC ou autre langage de faire de la simulation de trajectoires de corps célestes sous l'effet de la gravité.

I - Présentation du problème

Les mouvements des corps célestes sont régis les lois de la *gravité*, force d'interaction entre les masses et dont la loi de NEWTON en est une très bonne représentation :

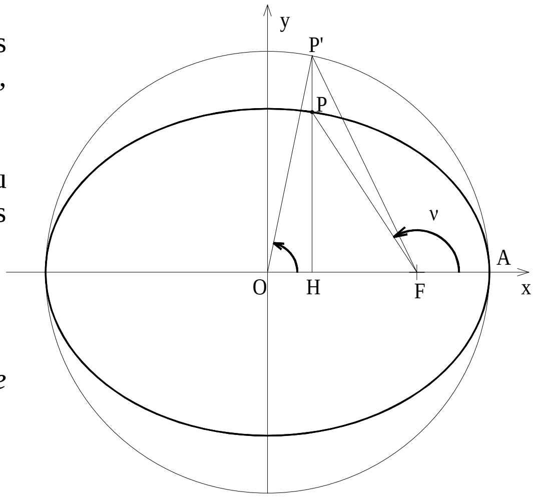
deux masses en présence s'attirent avec une force
- proportionnelle à leurs masses respectives
- inversement proportionnelle au carré de leur distance.

$$F = -G\frac{M_1 M_2}{r^2}$$

G constante de la gravitation : 6,672 . 10^{-11} N . m^2 . kg^{-2}

En l'absence de perturbation, les mouvements de deux corps en interaction gravitationnelle sont régis par la loi de Newton et s'expriment par les trois lois de Kepler.

- Première loi : sous l'action d'une force d'attraction, un corps céleste se déplace dans le champ d'attraction de l'autre corps céleste, suivant une conique (ellipse, parabole ou hyperbole).



La solution analytique est alors connue, c'est la résolution du problème du mouvement d'un point soumis à une force centrale et les trajectoires sont du type :

$$r = \frac{a(1-e^2)}{1+e\cos\nu}$$

r distance au foyer de l'ellipse, a *demi grand axe* de l'ellipse, ν *anomalie vraie*, angle entre le grand axe et le rayon vecteur, et

$$e = \frac{c}{a}$$

$c^2 = a^2 - b^2$, b demi petit axe. ***e*** est l'*excentricité* de l'orbite et correspond à l'aplatissement de la courbe. Si ***e*** = 0, l'ellipse devient est un cercle, ***e*** = 1 une parabole et ***e*** > 1 une hyperbole. ν est une fonction du temps.

- 2ème loi : l'aire balayée par le rayon vecteur varie proportionnellement au temps :

$$r^2.\frac{dv}{dt} = C^{te}$$

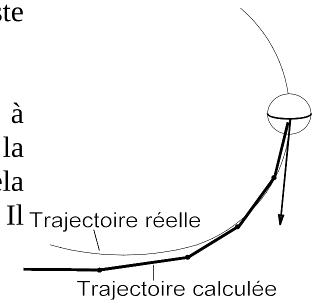
- 3ème loi : les carrés des durées de révolution d'un corps céleste autour d'un astre central sont proportionnels au cube du grand axe

$$\frac{a^3}{P^2} = C^{te}$$

Le problème se complique dès que l'on met en présence des corps supplémentaires qui ne permettent plus des solutions analytiques excepté dans des conditions extrêmement particulières. On a à traiter ce que l'on appelle le ***problème des N corps.*** On peut écrire pour chaque corps les équations représentant les forces agissantes sous forme d'équations différentielles du second degré (accélération) fonction des distances entre les corps.

La difficulté peut être en partie palliée si les corps supplémentaires ont des actions faibles par rapport aux deux premiers venus. On considère alors leurs actions comme des perturbations, et des méthodes de calculs existent dans ces cas là, d'ailleurs extrêmement difficiles à mettre en oeuvre sans le secours de l'informatique.

Sinon il reste la méthode itérative d'intégrer ces équations pas à pas. Il existe différentes méthodes dont on verra le principe de quelques unes.



Le calcul se fait pas à pas, c'est à dire que l'on calcule et positionne les objets à intervalles réguliers. De plus pour simplifier et accélérer le calcul, on suppose que la force exercée sur un corps par tous les autres est constante entre deux itérations. Cela signifie, qu'entre deux positions de calcul, la trajectoire est un segment de droite. Il faudra donc prendre quelques précautions :

- un pas d'échantillonnage le plus petit possible pour que l'approximation reste vraie, mais aussi pas trop court pour que l'évolution du système ne demande pas des temps de calcul démentiel
- choisir l'algorithme d'itération avec soins pour ne pas sortir immédiatement d'une trajectoire réaliste.

Conditions de départ

Quel que soit le mode de calcul choisi, il faut connaître la configuration des corps à un certain moment : position et vitesse, ce que l'on dénomme *conditions initiales*. A un instant dénommé t_0 on doit connaître la position et la vitesse de chacun d'eux (3 valeurs pour la position, et 3 pour la vitesse). Si le phénomène se passe dans un plan, comme le cas de la force centrale d'un corps unique, ou de deux corps, 4 valeurs initiales suffisent par élément.

II - Base de calcul et itération

A - Force centrale

a) Méthode *analytique*.

La solution analytique est alors connue, c'est la résolution du problème du mouvement d'un point soumis à une force centrale et les trajectoires dont du type :

$$r = \frac{a(1-e^2)}{1+e\cos\nu}$$

e est l'excentricité de l'orbite et a demi grand axe.
ν l'anomalie vraie est fonction du temps (figure 1).

Si P est la période de la planète, sa vitesse angulaire moyenne ou *mouvement moyen* est $\Theta = 2\pi / P$.

On appelle *anomalie moyenne* l'angle $\boldsymbol{M} = \Theta.(t - t_0)$, t_0 est l'instant du passage au *périgée* (distance la plus courte, *périhélie* dans le cas d'une planète autour du soleil).

L'*anomalie excentrique u* (voir figure 1) est reliée à l'anomalie moyenne par la relation

$$u = M + e.\sin u$$

qui n'a pas de solution analytique, mais se résoud par itération en prenant comme départ $u_0 = \boldsymbol{M}$.

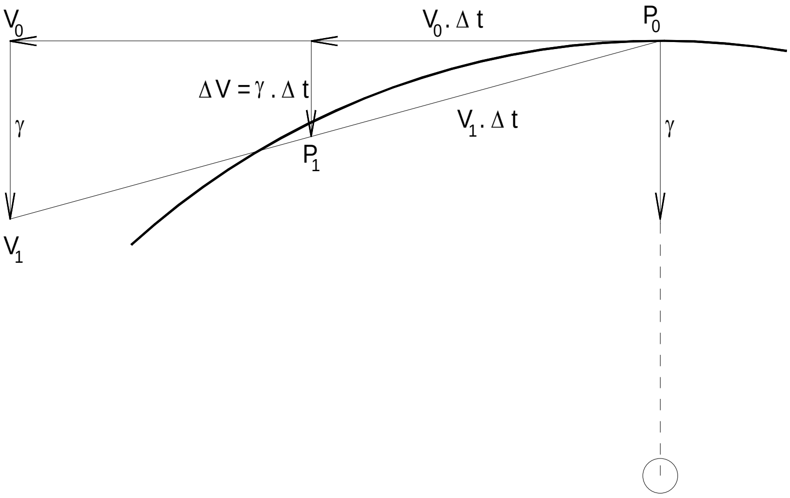
L'*anomalie vraie* ν (figure 1) est reliée à l'anomalie excentrique par la relation

$$\tan\frac{\nu}{2} = \sqrt{\frac{1+e}{1-e}}.\tan\frac{u}{2}$$

Le problème informatique revient à tracer une belle courbe dont on connaît tous les éléments : à chaque instant on calcul l'angle de position correspondant en passant de ***M*** à *u* puis à ν, calculer le rayon vecteur, en déduire les coordonnées cartésiennes.

b) Méthode de la *tangente*.

Pendant un instant Δt, on suppose que la vitesse est constante, on a alors la position à l'instant $t' = t+\Delta t$, on calcule la nouvelle vitesse à t' par $\Delta v / \Delta t = \Gamma$, et l'on recommence.



Ce procédé brutal ne peut donner une courbe satisfaisante que si le pas est très petit et si la courbure de l'orbite n'est pas trop forte. Sinon, la courbe obtenue est beaucoup moins fermée que la courbe réelle.

c) Méthode *développement de Taylor*

Pour améliorer le calcul de la vitesse et de l'accélération à chaque pas, on peut exprimer sa position en développement de Taylor en fonction de *t*.

$$f(x) = f(x_0) + \frac{x_0}{1!}.f'(x_0) + \frac{x_0^{\ 2}}{2!}.f''(x_0) + \ldots$$

Où la fonction f s'écrit vectoriellement par la position $\mathbf{X}_t$ d'un mobile, ainsi que sa vitesse$\mathbf{V}_t$, dérivée première, et la dérivée seconde l'accélération $\mathbf{\Gamma}_t$) :

$$X_{t+\Delta t} = X_t + \Delta t.V_t + \frac{\Delta t^2}{2!}.\Gamma_t + \frac{\Delta t^3}{3!}.\frac{(\Gamma_t - \Gamma_{t-\Delta t'})}{\Delta t'}$$

$$V_{t+\Delta t} = V_t + \Delta t.\Gamma_t + \frac{\Delta t^2}{2!}.\frac{(\Gamma_t - \Gamma_{t-\Delta t'})}{\Delta t'}$$

Le vecteur Γ_t est l'accélération due à la force centrale qui varie pour chaque position.

B - Cas des N corps

a) *Méthode de la tangente*

La méthode est identique à la méthode de la force centrale, mais il faut calculer la résultante de toutes les forces des autres corps et s'il y a lieu les forces dues à la rotation.

b) **Méthode développement de Taylor**

Même raisonnement que pour une force centrale, avec ici une expression plus complexe de l'accélération pour chacun des corps. Le centre de gravité du système n'étant pas obligatoirement immobile, il faut pour que le dessin reste dans l'écran, se ramener au référentiel centre de gravité.

c) **Méthode de Runge Kutta**

On exprime la position de tous les corps par un seul vecteur ***U*** à 6 x N éléments : 2x3 pour positions et vitesse p_x, p_y, p_z, v_x, v_y, v_z,. La fonction vectorielle f est la dérivée de *U* par rapport au temps. Donc *f* a pour composantes v_x, v_y, v_z, g_x, g_y, g_z, la vitesse et l'accélération.

On applique les 5 transformations suivantes pour obtenir la position suivante $\boldsymbol{U}_1$ au temps $t+h$

$$K_1 = h.\,f\,(t, U)$$
$$K_2 = h.\,f\,(t + h/2, \frac{K_1}{2})$$
$$K_3 = h.\,f\,(t + h/2, \frac{K_2}{2})$$
$$K_4 = h.\,f\,(t + h, K_3)$$
$$U_1 = U + \frac{(K_1 + 2.K_2 + 2.K_3 + K_4)}{6}$$

La méthode quoique plus lourde en calcul donne des résultats beaucoup plus précis.

C - Structure du programme

Quelle que soit la méthode utilisée, il faut se donner pour tous les corps mis en présence

- les masses, les positions et les vitesses à un instant t_0 que l'on prendra pour origine,

- se définir les facteurs d'échelle des distances et de temps, sauf si l'on s'arrange à travailler en coordonnées écran,

- dessiner la position initiale des corps

- entrer dans l'itération en faisant le calcul des positions et vitesses à la prochaine étape, effacer s'il y a lieu les anciennes positions (ou laisser des traces), tracer les nouvelles et recommencer

- prévoir des tests d'arrêt (dans le temps, dans l'espace...),

- prévoir si cela est nécessaire des changement du pas d'itération : diminuer quand les corps sont très proches (interaction forte), augmenter dans le cas contraire,

- afficher quelques résultats dans un angle de l'écran pour mieux saisir la situation.

III - Bibliographie

Meus : Calculs astronomiques pour amateurs
Bouige S. : Calculs Astronomiques pour Amateurs adapté à l'emploi d'u calculateur électronique de poche Masson 1980.
Guy Sérane : Astronomie et ordinateur Dunod 1987
Mineur H. : Techniques de calcul numérique Librairie Technique Ch Béranger 1952, page 365.
Bass : Cours de mathématique Tome II, Masson Ed.

FORCE CENTRALE - SOLUTION ANALYTIQUE
TURBO BASIC

```
'
' Astronomie et Informatique
' Trajectoire des PlanÞtes
' force centrale - methode analytique
' Programme : simgrav0.bas
' Ph Merlin 10/01/1990
'
'
' entree des paramÞtres initiaux : période, demi-grand axe, ellipticité
'
SCREEN 12
DEFDBL A-H,O-Z
INPUT "Periode (en jours)    : ";P
INPUT "Demi grand axe (u.a.) : ";A
INPUT "Ellipticite           : ";E
INPUT "Pas calcul (jours)    :";PAS
PRINT "T=0 : passage perigee"
INPUT "   temps depart       : ";T0
INPUT "   temps de fin       : ";TF
CLS
' coordonnees du centre
XC = 320 : YC = 240
ech = yc/(a*e+a)
' trace du foyer
LINE (xc-2,yc-2)-(xc+2,yc+2)
LINE (xc-2,yc+2)-(xc+2,yc-2)
LOCATE 1,5,0 : PRINT"appuyer sur une touche s.v.p."
' attente pour depart
A:
IF INKEY$ = "" GOTO A:
‘ écriture d’une ligne d’espaces pour effacer le texte
LOCATE 1,5,0 : PRINT"                              "
PI = 3.14152654#
VR = 2! * PI / P
'
' label pour le retour aux itérations sur le temps
'
B:
' calcul anomalie moyenne
ANM = (T + T0) * VR
' calcul anomalie excentrique par iteration
ANE0 = ANM : NK = 0
E: : ‘label de l’itération du calcul de l’anomalie excentrique
ANE1 = ANM + E * SIN(ANE0)  : NK = NK+1 : IF NK > 100 THEN END
IF ABS(ANE1 - ANE0) > .0000001 THEN ANE0 = ANE1 : GOTO E:
' calcul anomalie vraie
Z = SQR((1+E) / (1-E)) * TAN(ANE1/2!)
ANV = 2! * ATN(Z)
' calcul rayon vecteur
R = A * (1 - E*E) / (1 + E * COS(ANV))
LOCATE 1,1:PRINT USING "T:####";T
' coordonnees du point
X = R * COS(ANV)
Y = R * SIN(ANV)
' trace du point
PSET(X*ech+XC,Y*ech+YC),6
' tests d'arret
T$ = INKEY$ : IF T$ <> "A" THEN : GOTO D:
C: : IF INKEY$ = "" GOTO C: : ‘ boucle d’attente
D: : ‘ label de saut
T = T + PAS : ‘incrémentation du temps
IF T$ = "F" OR T$ = "f" OR T > TF THEN END ELSE GOTO B:
end
```

```
'
' Astronomie et Informatique
' Simulation des trajectoires des Planetes
' Ph Merlin 11/01/2001
' Programme : simgrav1.bas
'
' Paramètres initiaux
'
DEFDBL A-H,O-Z          : ' double précision pour les calculs
SCREEN 12               : ' écran VGA 640x480
XC = 320 : YC = 240     : ' position de l'origine
UA = 1.4959787E+11      : ' unité astronomique
ECH = 640./3./UA        : ' échelle du dessin - largeur = 3.0 u.a.
SM = 1.989E+30          : ' masse solaire
GK = 6.672E-11          : ' constante de la gravitation
' entrée des données
PRINT : PRINT "Simulation de la gravité dans le système solaire"
INPUT "Position départ (x et y) en U.A.  :";X0,Y0
INPUT "Vitesse départ (vx et vy) en km/s :";VX0,VY0
INPUT "Masse centrale en masses solaires :";SOLM
INPUT "Incrément du temps en secondes    :";DT
CLS
' affichage des données
LOCATE 1,20,0 : PRINT USING "M=##.## x0=##.## y0=##.##";SOLM,X0,Y0;
PRINT USING " vx0=##.## vy0=##.## DT=#######";VX0,VY0,DT;
' Unités de tracé et calcul
SOLM = SOLM * SM        : ' en kilogrammes
X0 = X0 * UA            : ' en mètres
Y0 = Y0 * UA            : ' en mètres
VX0 = VX0 * 1000.       : ' en km / s
VY0 = VY0 * 1000.       : ' en km / s
'
' initialisation dessin
'
' centre
LINE (XC+2,YC+2)-(XC-2,YC-2)
LINE (XC-2,YC+2)-(XC+2,YC-2)
PSET(X0*ECH+XC,Y0*ECH+YC),2
LOCATE 2,16,0:PRINT "Pour commencer appuyer sur une touche s.v.p."
A: : IF INKEY$ = "" GOTO A  ELSE T$ = INKEY$
LOCATE 2,16,0:PRINT "                                            "
LOCATE 2,1,0:PRINT "Fin='F'"
ITER = 1
'
' itération
'
B:
'
' calcul de l'accélération
D0 = SQR(X0*X0+Y0*Y0)   : ' distance Soleil - planète
GAM0 = -GK*SOLM/(D0*D0) : ' accélération
SALPHA0 = Y0/D0         : ' sinus de l'angle du vecteur Soleil - planète
CALPHA0 = X0/D0         : ' cosinus de l'angle du vecteur Soleil - planète
DX0 = GAM0*CALPHA0      : ' composante x de l'accélération
DY0 = GAM0*SALPHA0      : ' composante y de l'accélération
VX1 = VX0 + DX0*DT      : ' composante x de la nouvelle vitesse
VY1 = VY0 + DY0*DT      : ' composante y de la nouvelle vitesse
X1 = X0 + DT*VX1        : ' nouvelle abscisse du point
Y1 = Y0 + DT*VY1        : ' nouvelle ordonnées du point
' mise à blanc du point précédent
PSET(X1*ECH+XC,Y1*ECH+YC)
PSET(X0*ECH+XC,Y0*ECH+YC),4
LOCATE 1,1,0 : PRINT USING "######.#j";ITER*DT/3600./24 : ITER = ITER+1
' transfer des données pour le pas suivant
X0 = X1 : Y0 = Y1
VX0 = VX1 : VY0 = VY1
T$ = INKEY$ : IF T$="F" OR T$="f" THEN PRINT "FIN" : END
GOTO B
END
```

```
'
' Astronomie et Informatique
' Simulation des trajectoires des Planetes
' Ph Merlin 11/01/2001
' Programme : simgrav2.bas
'
' Paramètres initiaux
'
DEFDBL A-H,O-Z           : ' double précision pour les calculs
SCREEN 12                : ' écran VGA 640x480
XC = 320 : YC = 240      : ' position de l'origine
UA = 1.4959787E+11       : ' unité astronomique
ECH = 640./3./UA         : ' échelle du dessin - largeur = 3.0 u.a.
SM = 1.989E+30           : ' masse solaire
GK = 6.672E-11           : ' constante de la gravitation
' entrée des données
PRINT : PRINT "Simulation de la gravité dans le système solaire"
INPUT "Position départ (x et y) en U.A.  :";X0,Y0
INPUT "Vitesse départ (vx et vy) en km/s :";VX0,VY0
INPUT "Masse centrale en masses solaires :";SOLM
INPUT "Incrément du temps en secondes    :";DT
CLS
' affichage des données
LOCATE 1,20,0 : PRINT USING "M=##.## x0=##.## y0=##.##";SOLM,X0,Y0;
PRINT USING " vx0=##.## vy0=##.## DT=#######";VX0,VY0,DT;
' Unités de tracé et calcul
SOLM = SOLM * SM         : ' en kilogrammes
X0 = X0 * UA             : ' en mètres
Y0 = Y0 * UA             : ' en mètres
VX0 = VX0 * 1000.        : ' en km / s
VY0 = VY0 * 1000.        : ' en km / s
'
' initialisation dessin
'
' centre
LINE (XC+2,YC+2)-(XC-2,YC-2)
LINE (XC-2,YC+2)-(XC+2,YC-2)
PSET(X0*ECH+XC,Y0*ECH+YC),2
LOCATE 2,16,0:PRINT "Pour commencer appuyer sur une touche s.v.p."
RET1: : IF INKEY$ = "" GOTO RET1  ELSE T$ = INKEY$
LOCATE 2,16,0:PRINT "                                            "
LOCATE 2,1,0:PRINT "Fin ='F'"
LOCATE 3,1,0:PRINT "Stop='S'"
ITER = 1
D0 = SQR(X0*X0+Y0*Y0)    : ' distance Soleil - planète
GAM0 = -GK*SOLM/(D0*D0)  : ' accélération
SALPHA0 = Y0/D0          : ' sinus de l'angle du vecteur Soleil - planète
CALPHA0 = X0/D0          : ' cosinus de l'angle du vecteur Soleil - planète
DX0 = GAM0*CALPHA0       : ' composante x de l'accélération
DY0 = GAM0*SALPHA0       : ' composante y de l'accélération
DX0 = 0. : DY0 = 0.
'
' itération
'
' calcul de l'accélération
ITER1:
D0 = SQR(X0*X0+Y0*Y0)    : ' distance Soleil - planète
GAM0 = -GK*SOLM/(D0*D0)  : ' accélération
SALPHA0 = Y0/D0          : ' sinus de l'angle du vecteur Soleil - planète
CALPHA0 = X0/D0          : ' cosinus de l'angle du vecteur Soleil - planète
DX1 = GAM0*CALPHA0     : ' composante x de l'accélération
DY1 = GAM0*SALPHA0     : ' composante y de l'accélération
VX1 = VX0+DT*DX0+DT*DT*(DX1-DX0)/(2*DT):' composante x de la nouvelle vitesse
VY1 = VY0+DT*DY0+DT*DT*(DY1-DY0)/(2*DT):' composante y de la nouvelle vitesse
X1 = X0+DT*VX1+DT*DT*DX1/2+DT*DT*DT*(DX1-DX0)/(6*DT)
Y1 = Y0+DT*VY1+DT*DT*DY1/2+DT*DT*DT*(DY1-DY0)/(6*DT)
' Tracé en blanc du nouveau point
PSET(X1*ECH+XC,Y1*ECH+YC)
' tracé en rouge du point précédent (couleur 4)
PSET(X0*ECH+XC,Y0*ECH+YC),4
for i =0 to 1000 : next i
LOCATE 1,1,0 : PRINT USING "######.#j";ITER*DT/3600./24 : ITER = ITER+1
' transfer des données pour le pas suivant
X0 = X1 : Y0 = Y1
VX0 = VX1 : VY0 = VY1
DX0 = DX1 : DY0 = DY1
T$ = INKEY$ : IF T$="F" OR T$="f" THEN PRINT "FIN" : END
IF T$="S" OR T$="s" GOTO SAUT1 ELSE GOTO ITER1
SAUT1: : T$ = INKEY$ : IF T$="" GOTO SAUT1
```

```
GOTO ITER1
END
```

```
'
' Astronomie et Informatique
' Simulation des trajectoires des Planetes
' Ph Merlin 11/01/2001
' Programme : simgrav3.bas méthode de la tangente - cas de N corps
'
' Paramètres initiaux
'
DEFDBL A-H,O-Z           : ' double précision pour les calculs
SCREEN 12                : ' écran VGA 640x480
XC = 320 : YC = 240      : ' position de l'origine
UA = 1.4959787E+11       : ' unité astronomique
ECH = 640./3./UA         : ' échelle du dessin - largeur = 3.0 u.a.
SM = 1.989E+30           : ' masse solaire
GK = 6.672E-11           : ' constante de la gravitation
' entrée des données
PRINT : PRINT "Simulation de la gravité dans le système solaire"
INPUT "Nombre de corps :";N
DIM X0(N),Y0(N),VX0(N),VY0(N),CPM(N),X1(N),Y(1),VX1(N),VY1(0),COL(N)
FOR I=1 TO N
  CLS
  PRINT "CORPS n° :",I
  INPUT "  Position départ (x et y) en U.A.  :";X0(I),Y0(I)
  INPUT "  Vitesse départ (vx et vy) en km/s :";VX0(I),VY0(I)
  INPUT "  Masse centrale en masses solaires :";CPM(I)
NEXT I
PRINT : INPUT "Incrément du temps en secondes    :";DT
INPUT "Centre de gravité O/N :";R$
CLS
' Unités de tracé et calcul
FOR I=1 TO N
  CPM(I) = CPM(I) * SM          : ' en kilogrammes
  X0(I) = X0(I) * UA            : ' en mètres
  Y0(I) = Y0(I) * UA            : ' en mètres
  VX0(I) = VX0(I) * 1000.       : ' en km / s
  VY0(I) = VY0(I) * 1000.       : ' en km / s
  COL(I) = 1+(I MOD 6)
NEXT I
' centre de gravité
P1: : IF R$="n" OR R$="N" GOTO P1
print x0(1),y0(1),x0(2),y0(2): print
XX=0. : YY=0. : VX=0. : VY=0. : CM=0.
FOR I=1 TO N
  XX=XX+X0(I)*CPM(I) : YY=YY+Y0(I)*CPM(I)
  VX=VX+VX0(I)*CPM(I) : VY = VY+VY0(I)*CPM(I)
  CM=CM+CPM(I)
NEXT I
XX=XX/CM : YY=YY/CM : VX=VX/CM : VY=VY/CM
FOR J=1 TO N
  X0(J)=X0(J)-XX : Y0(J)=Y0(J)-YY : VX0(J)=VX0(J)-VX : VY0(J)=VY0(J)-VY
NEXT J
'
' initialisation dessin
'
' retour pour zoom
LZOOM:
CLS
' tracé du centre et des positions des corps
LINE (XC+2,YC+2)-(XC-2,YC-2)
LINE (XC-2,YC+2)-(XC+2,YC-2)
FOR I=1 TO N :PSET(X0(I)*ECH+XC,Y0(I)*ECH+YC),2 : NEXT I
'
LOCATE 2,16,0:PRINT "Pour commencer appuyer sur une touche s.v.p."
RET2: : IF INKEY$ = "" GOTO RET2  ELSE T$ = INKEY$
' écriture des options
LOCATE 2,16,0:PRINT "                                            "
LOCATE 2,1,0:PRINT "F Fin"
LOCATE 3,1,0:PRINT "z Zoom+"
LOCATE 4,1,0:PRINT "Z Zoom-"
ITER = 1
'
' itération
'
' calcul des composantes de l'accélération de chaque corps
LABITER:
 FOR I=1 TO N
  GAMX=0. : GAMY=0.
  FOR J=1 TO N
   IF I=J GOTO SAUT1 : ' on saute pour le corps lui-même
```

```
    ' distance Soleil - planète
    D0 = SQR((X0(I)-X0(J))^2+(Y0(I)-Y0(J))^2)
    GAM = -CPM(J)*GK/(D0*D0)  : ' accélération
    SALPHA = (Y0(I)-Y0(J))/D0 : ' sinus de l'angle du vecteur Soleil - planète
    CALPHA = (X0(I)-X0(J))/D0 : ' cosinus de l'angle du vecteur Soleil - te
    DX0 = GAM*CALPHA          : ' composante x de l'accélération
    DY0 = GAM*SALPHA          : ' composante y de l'accélération
    GAMX=GAMX+DX0 : GAMY=GAMY+DY0 : ' sommation
SAUT1:
  NEXT J
  VX1(I) = VX0(I) + GAMX*DT  : ' composante x de la nouvelle vitesse
  VY1(I) = VY0(I) + GAMY*DT  : ' composante y de la nouvelle vitesse
  X1(I) = X0(I) + DT*VX1(I)  : ' nouvelle abscisse du point
  Y1(I) = Y0(I) + DT*VY1(I)  : ' nouvelle ordonnées du point
' tracé du nouveau point en blanc
  PSET(X1(I)*ECH+XC,Y1(I)*ECH+YC)
' mise à la couleur de la position précédente
  PSET(X0(I)*ECH+XC,Y0(I)*ECH+YC),COL(I)
NEXT I
' écriture temps écoulé
LOCATE 1,1,0 : PRINT USING "######.#j";ITER*DT/3600./24 : ITER = ITER+1
' transfer des données pour le pas suivant
FOR K=1 TO N
  X0(K)  = X1(K)  : Y0(K)  = Y1(K)
  VX0(K) = VX1(K) : VY0(K) = VY1(K)
NEXT K
' test de touche
T$ = INKEY$ : IF T$="F" OR T$="f" THEN PRINT "FIN" : END
IF T$ = "z" then ECH = ECH*2 : goto LZOOM
IF t$ = "Z" then ECH = ECH/2 : goto LZOOM
IF T$="S" OR T$="s" goto SAUT2 else GOTO LABITER
SAUT2: : if inkey$ = "" goto SAUT2
goto LABITER
END
```