Méthodes informatiques pour physiciens

introduction à C++ et
résolution de problèmes de physique par ordinateur

Leçon # 10 : Equations Différentielles Ordinaires
Muovements périodiques

Alessandro Bravar Alessandro.Bravar@unige.ch
tél.: 96210 bureau: EP 206

assistants

Johanna Gramling Johanna.Gramling@unige.ch
tél.: 96368 bureau: EP 202A

Mark Rayner Mark.Rayner@unige.ch
tél.: 96263 bureau: EP 219

http://dpnc.unige.ch/~bravar/C++2015/L10
pour les notes du cours, les exemples, les corrigés, …

Plan du jour #10

Récapitulatif et corrigé de la leçon #9

Résolution numérique des équations différentielles – rappels

Le pendule

Les orbites

Constructions d'une classe `Planete` (Système Solaire)

Textes conseilles pour l'analyse numérique

M.-Y. Bachmann et al. (CRM)
Méthodes numériques

référence — W.H. Press et al.
Numerical Recipes

voir aussi — Feynman / Leighton / Sands
Le cours de physique de Feynman vol. 1 ch. 9

Récapitulatif de la leçon #9

Programmation orienté objets

Les classes

```
class
```

Données et Méthodes

Données membres

```
public
private
protected
```

Constructeurs et destructeurs

Fonctions d'accès

Surcharge des operateurs

La bibliothèque STL

Résolution numérique des équations différentielles

Considérons l'équation différentielle $\frac{dy}{dx} = f(x,y) \qquad y(x_0) = y_0$

Après intégration avec la Formule du Rectangle (L5) on obtient

$$y(x_0 + h) = y_0 + \int_{x_0}^{x_0+h} f(x,y)\,dx \approx y_0 + f(x_0, y_0) \times h$$

Cette formule correspond à la méthode d'intégration d'Euler pour l'intégration des équations différentielles.

En notation vectorielle (une équation différentielle d'ordre p peut être réduite à un système de p équations différentielles de 1er ordre), la solution de

$$\frac{d\mathbf{r}(t)}{dt} = \mathbf{f}(\mathbf{r}(t), t)) \qquad \mathbf{r}(t) = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix} \qquad \mathbf{f}(\mathbf{r}(t), t)) = \begin{pmatrix} v(t) \\ a(t) = F/m \end{pmatrix}$$

est donnée selon la méthode d'Euler par :

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \mathbf{f}(\mathbf{r}(t), t) \times \Delta t$$

avec l'algorithme numérique suivant :

et conditions initiales

$$x_0 = x(t=0)$$
$$v_0 = v(t=0)$$

$$x_{n+1} = x_n + v_n \times \Delta t$$
$$v_{n+1} = v_n + \frac{F(x_n, v_n)}{m} \times \Delta t$$

Méthode de Euler modifié → Méthode de Runge

Pour améliorer la précision du calcul de l'intégrale, on a introduit la Formule du Trapèze (on évalue la fonction en deux points). On peut procéder de la même façon pour l'intégration des équations différentielles et modifier la méthode :

$$y_1 = y_0 + \frac{1}{2}\left[f\left(x_0, y_0\right) + f\left(x_1, y_1^{(0)}\right)\right] \times h$$

où $y_1^{(0)}$ a été évaluée avec la méthode d'Euler en $x_1 = x_0 + h$.

La question est toujours où calculer la dérivée !!!

au début, à la fin, ou au milieu de l'intervalle h ?

Après substitution

$$y_1 = y_0 + \frac{1}{2}\left[f\left(x_0, y_0\right) + f\left(x_0 + h, y_0 + f\left(x_0, y_0\right) \times h\right)\right] \times h$$

Au lieu de calculer *f* en x_i et $x_i + h$, on peut calculer f en $x_i + h/2$:

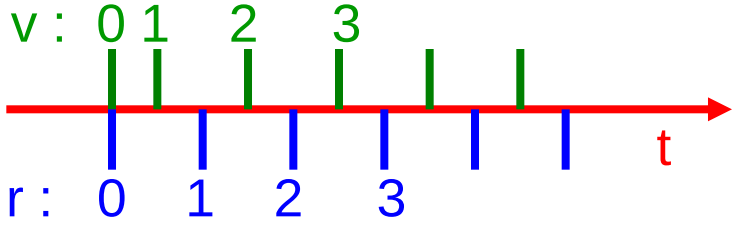
$$y_1 = y_0 + f\left(x_0 + \frac{1}{2}h, y_0\right) \times h$$

Cette méthode est connue comme méthode de Runge de seconde ordre.

L'algorithme d'intégration devient :

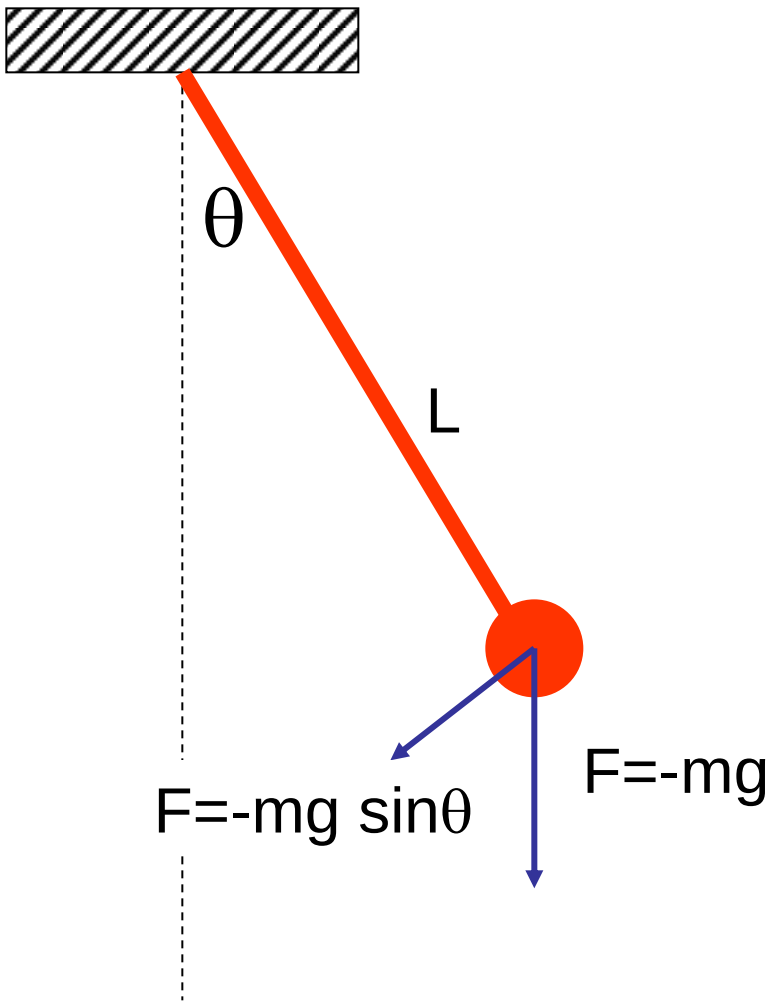
$$v_{n+1} = v_n + \frac{F\left(x_i, v_i\right)}{m} \times \Delta t$$

$$x_{n+1} = x_n + v_{n+1} \times \Delta t$$



mais avec la condition initiale $v_0 = v_0 - \frac{F(x_0, v_0)}{m} \times \frac{\Delta t}{2}$ ou $v_1 = v_0 + \frac{F(x_0, v_0)}{m} \times \frac{\Delta t}{2}$

Le pendule



«équation du mouvement»
(en coordonnées polaires)

$$\frac{\mathrm{d}^2\vartheta(t)}{\mathrm{d}t^2} = -\frac{g}{L}\sin\vartheta(t)$$

approximation petites oscillations ($\theta < 10^0$)

$$\frac{\mathrm{d}^2\vartheta(t)}{\mathrm{d}t^2} \approx -\frac{g}{L}\vartheta(t)$$

avec la solution

$$\vartheta(t) = \vartheta_{max}\cos(\omega_0 t + \vartheta_0)$$

et le période d'oscillation

$$\omega_0 = \frac{2\pi}{T_0} \qquad T_0 = 2\pi\sqrt{\frac{L}{g}}$$

θ_{max} et θ_0 sont donnés
par les conditions initiales

Résolution numérique – Méthode d'Euler (1768)

D'abord on essaye avec la Méthode d'Euler.

À un moment donné t, le pendule a une certaine vitesse angulaire ω et un angle θ. Quelles sont la vitesse et l'angle un peu plus tard au temps $t + \Delta t$?

A chaque instant t, la position au temps $t + \Delta t$ est (si Δt est suffisamment petit) :

$$\vartheta(t+\Delta t) = \vartheta(t) + \omega(t)\cdot\Delta t \qquad \left[x(t+\Delta t) = x(t) + v(t)\cdot\Delta t\right]$$

Pour obtenir la vitesse ω au temps $t + \Delta t$, il nous faut connaître comment la vitesse change, c'est-à-dire l'accélération angulaire α. L' accélération est donnée par les lois de la dynamique, c'est-à-dire la loi du mouvement (eq. de Newton)

$$\omega(t+\Delta t) = \omega(t) + \alpha(t)\cdot\Delta t$$

cinématique

où $\alpha = -g/L\theta$ (loi de la dynamique), donc

$$\omega(t+\Delta t) = \omega(t) - g/L\cdot\vartheta(t)\cdot\Delta t$$

dynamique

et on obtient l'algorithme suivant

$$\vartheta_{n+1} = \vartheta_n + \omega_n\cdot\Delta t$$
$$\omega_{n+1} = \omega_n - g/L\cdot\vartheta_n\cdot\Delta t$$

où θ_0 et ω_0 sont donnés par les conditions initiales

[même procédure que pour le mouvement des projectiles]

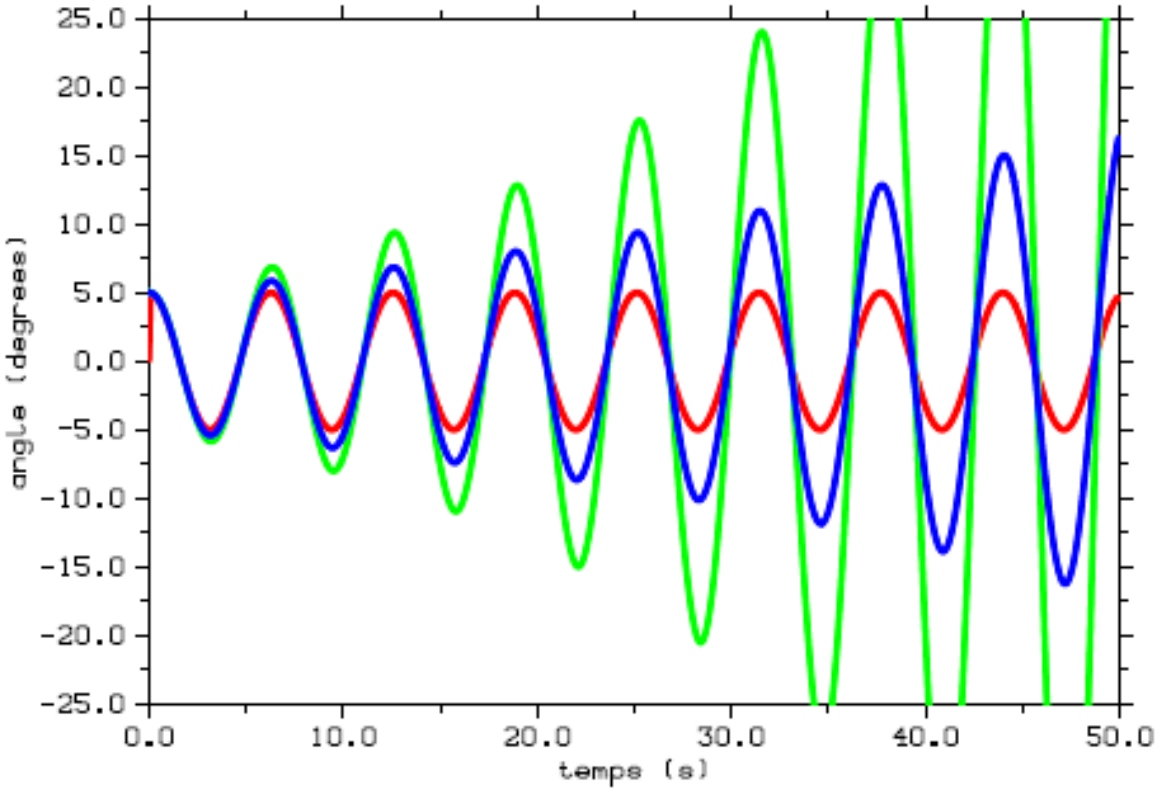
Oscillations petits angles – méthode d'Euler

Pour vérifier la méthode on compare la solution exacte avec la solution numérique.

$\theta_0(t=0) = 5^0$
$\omega_0(t=0) = 0$
$g/L = 1$

sol. exacte
$\Delta t = 0.1$
$\Delta t = 0.05$

voir Pendule_Euler.cpp



L'amplitude de l'oscillation augmente avec le temps et l'énergie n'est pas conservée ! tandis que la période d'oscillation ne change pas.
Ceci est dû aux approximations faites pendant l'intégration numérique de l'équation du mouvement : à chaque itération on cumule des erreurs de troncation qui se cumulent.
On peut essayer de Δt plus petits, mais ça n'est pas la bonne voie à suivre.

Où est le problème ?

En général, pour résoudre l'équation du mouvement

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = v(t)$$
$$\frac{\mathrm{d}v(t)}{\mathrm{d}t} = a\big(v(t), x(t)\big) = \frac{1}{m}F(t)$$

on doit calculer les dérivées de *r* et *v* (théorème de Taylor, i.e. $\exists\ \zeta$ t.q. $t < \zeta < t+\Delta t$)

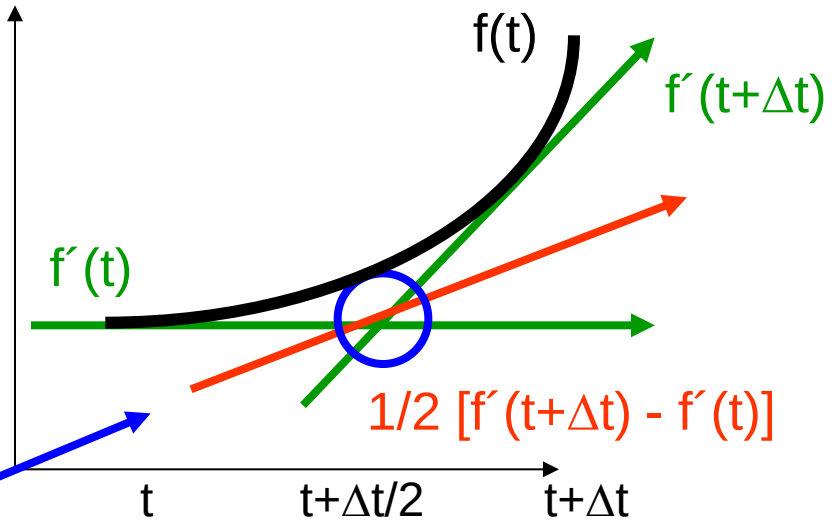
$$f'(t) = \frac{f(t+\Delta t) - f(t)}{\Delta t} - \frac{1}{2} f''(\zeta)\Delta t$$

terme ignoré
erreur de troncation

A chaque itération on cumule l'erreur de troncation et la solution *diverge* ;
l'énergie n'est plus conservée.

La question est donc où (ou quand) calculer la dérivée (vitesse, accélération) ?
au début, à la fin, au milieu
de l'intervalle temporel Δt ?

L'amélioration consiste à utiliser la vitesse (dérivée) calcule au *milieu*
et la position au début de l'intervalle Δt.



Implémentation de la méthode de Runge

vitesse calculée au milieu (entre t et Δt)

angle calculée à la fin de l'intervalle Δt

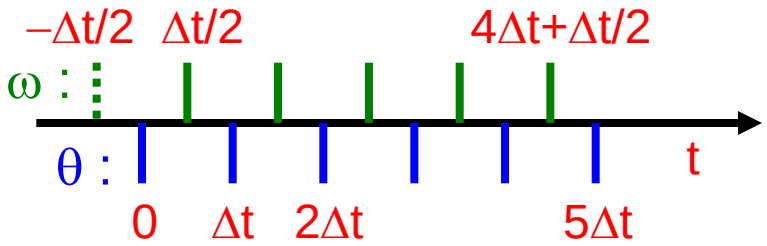
conditions initiales

$$\omega(t+\Delta t/2)=\omega(t-\Delta t/2)+\alpha(t)\cdot\Delta t$$
$$\vartheta(t+\Delta t)=\vartheta(t)+\omega(t+\Delta t/2)\cdot\Delta t$$
$$\vartheta(0)=\vartheta_0;\qquad \omega(0)=\omega_0$$

Le calcul de la vitesse est décalé de $\Delta t/2$ par rapport au calcul de l'angle :

$$\omega(t+\Delta t/2)=\frac{\vartheta(t+\Delta t)-\vartheta(t)}{\Delta t}+o(\Delta t^2)$$

ω : t–$\Delta t/2$, t+$\Delta t/2$, t+3$\Delta t/2$ … ; θ : t, t+Δt, t+2Δt …



On connaît $\omega(t=0)$ mais pas $\omega(-\Delta t/2)$ ou $\omega(\Delta t/2)$!
Pour commencer l'itération nous pouvons choisir

$\omega(-\Delta t/2) = \omega(0) - \alpha(0)\Delta t/2$ (non physique, mais …)

(i.e. utiliser la Méthode de Euler pour calculer ω en $-\Delta t/2$)
Cette méthode de Runge est d'ordre supérieur à Δt ($o(\Delta t^2)$) que la méthode d'Euler ($o(\Delta t)$).
Donc on obtient ainsi une approximation meilleure de la solution avec l'algorithme suivant :
A noter que d'abord on calcule la vitesse, puis l'angle.

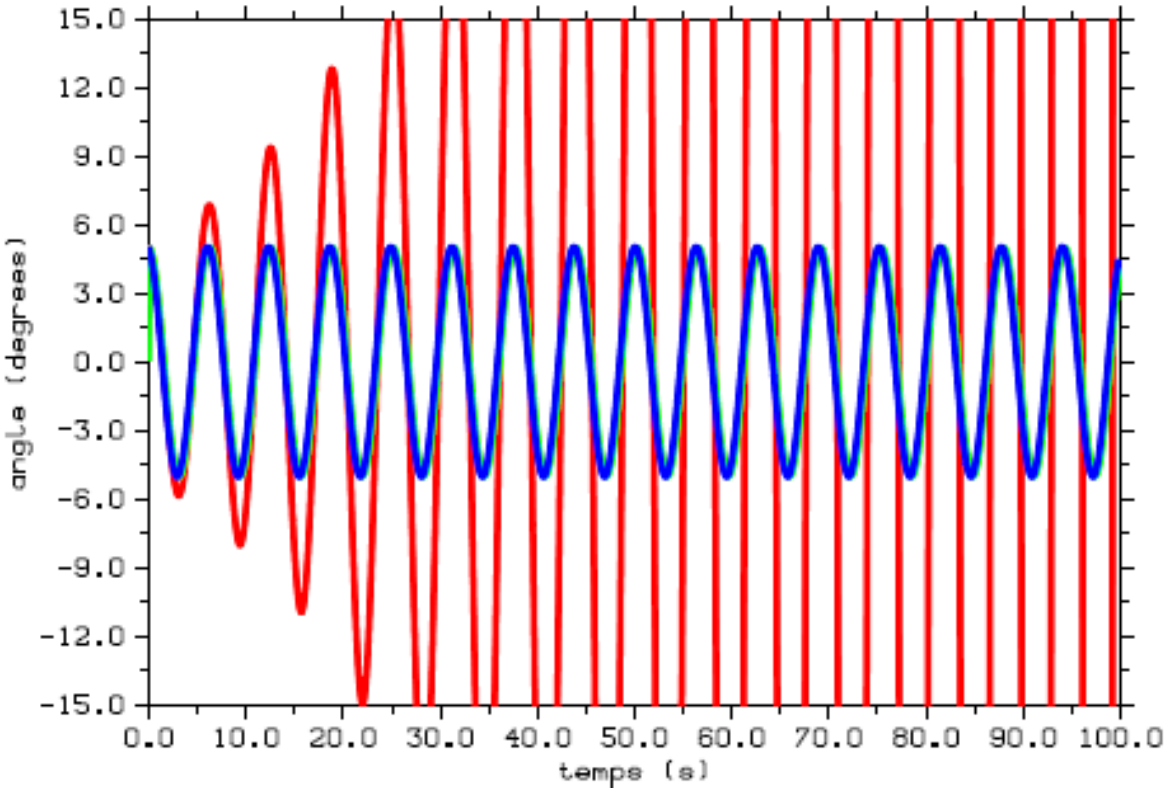
$$\omega_0=\omega(0)-\alpha(0)\cdot\Delta t/2$$
$$\alpha_n=\alpha(x_n,v_n)$$
$$\omega_{n+1}=\omega_n+\alpha_n\cdot\Delta t$$
$$\theta_{n+1}=\theta_n+\omega_{n+1}\cdot\Delta t$$

Oscillations petits angles – méthode de Runge

$\theta_0(t=0) = 5^0$
$\omega_0(t=0) = 0$
$g/L = 1$

$\Delta t = 0.1$

Euler
Runge
sol. exacte



voir Pendule_Runge.cpp

Avec la Méthode de Runge on trouve un accord parfait avec la solution exacte !

On peut donc utiliser cette méthode pour résoudre n'importe quel problème, pour commencer p.ex. l'équation du pendule physique (avec le sinus) sans l'approximation «petits angles».

Algorithmes d'Euler et Runge

```
const int steps = 5000;
double dt = 0.01;
double angle[steps], omega[steps], time[steps];
```

nous avons introduit des tableaux position, vitesse et temps pour dessiner la trajectoire avec DISLIN

méthode de Euler

```
angle[0] = 5.;    //angle initial
omega[0] = 0.;    //vitesse initiale
time[0] = 0.;

for (int i=0; i<steps-1; i++) {
  angle[i+1] = angle[i] + omega[i]*dt;
  omega[i+1] = omega[i] - angle[i]*dt;
  time[i+1]  = time[i] + dt;
}
```

conditions initiales

algorithme

grands angles :

remplacez `angle[i]` avec `sin(angle[i])`

méthode de Runge

```
angle[1] = 5.;    //angle initial
omega0   = 0.;    //vitesse initiale
omega[0] = omega0 + angle[0]*dt/2;
time[0]  = 0.;

for (int i=0; i<steps-1; i++) {
  omega[i+1] = omega[i] - angle[i]*dt;
  angle[i+1] = angle[i] + omega[i+1]*dt;
  time[i+1]  = time[i] + dt;
}
```

conditions initiales

algorithme

voir Pendule_Runge.cpp

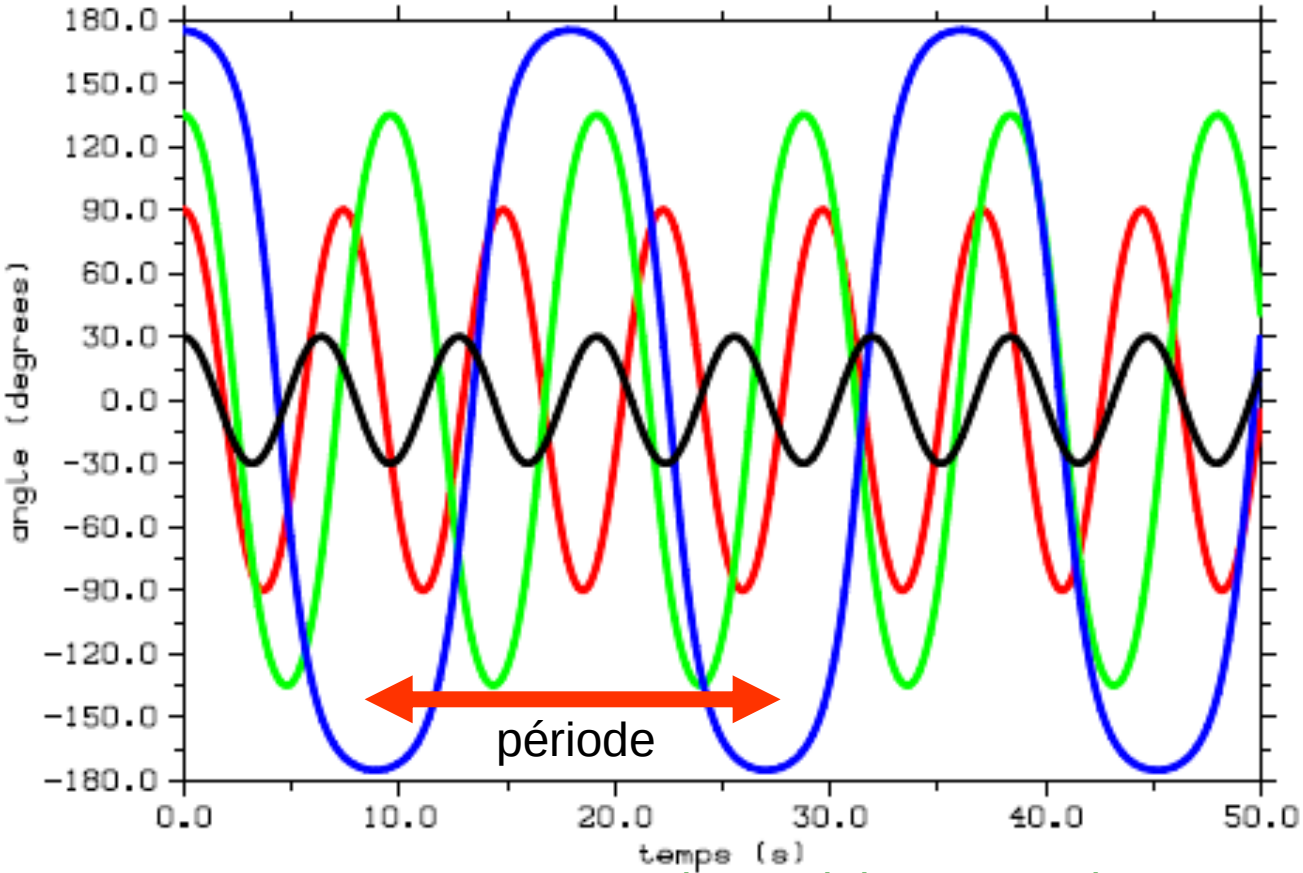
Oscillations grands angles – Méthode de Runge

pendule physique

$\theta_0(t=0) = 30^0$
$\theta_0(t=0) = 90^0$
$\theta_0(t=0) = 135^0$
$\theta_0(t=0) = 175^0$

$\omega_0(t=0) = 0$
$g/L = 1$

$\Delta t = 0.01$



voir Pendule_Runge_large.cpp

Pour les oscillations de grandes amplitudes (angles, $\theta_0 > 10^0$) on ne peut plus utiliser l'approximation petites oscillations (petits angles) et on doit résoudre l'équation du mouvement exact.

A noter : la période augmente avec l'amplitude
la trajectoire n'est plus sinusoïdale

* Période d'oscillation

L'équation du pendule sans approximation est plus difficile à résoudre. Nous savons que le mouvement est toujours périodique. On peut obtenir la période sans résoudre l'équation pour θ(t) avec la conservation de l'énergie.

$$E_{TOT} = E_{CIN} + U_{POT}$$

conservation de l'énergie

$$-mgL\cos\vartheta_{\max} = \frac{1}{2}mL^2\omega^2 - mgL\cos\vartheta$$

$$\omega = \frac{d\vartheta}{dt} = \sqrt{\frac{2g}{L}\left(\cos\vartheta - \cos\vartheta_m\right)}$$

$$dt = \frac{d\vartheta}{\sqrt{\frac{2g}{L}\left(\cos\vartheta - \cos\vartheta_{\max}\right)}}$$

période d'oscillation

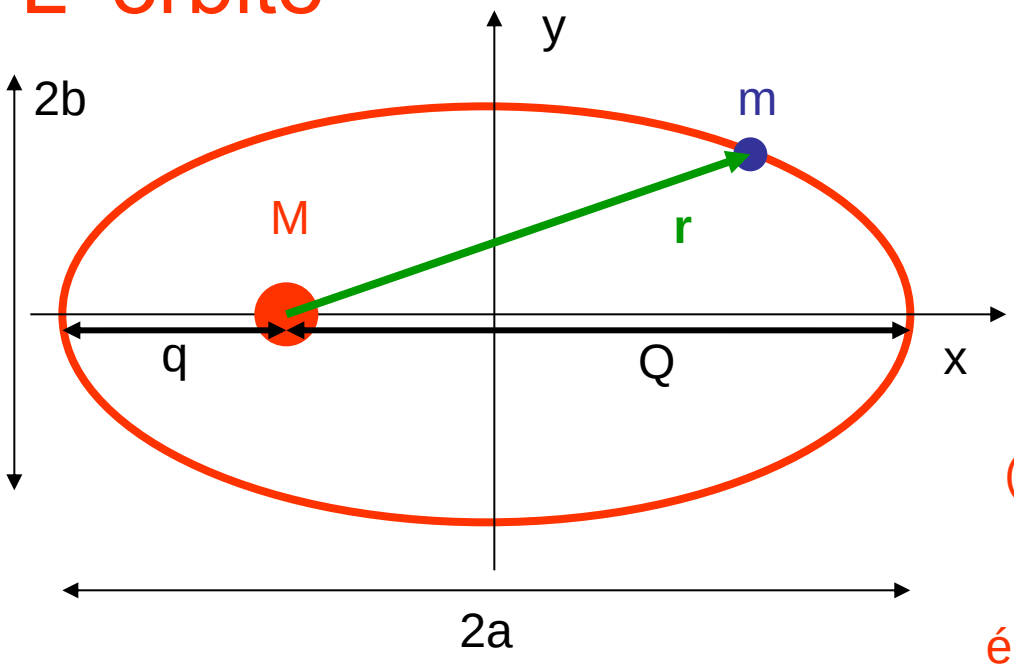
intégrale résolue avec une technique d'intégration numérique

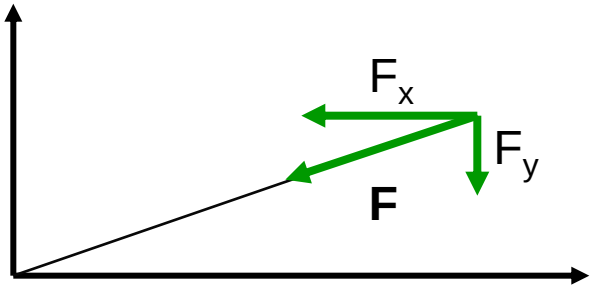
$$T = 4\sqrt{\frac{L}{2g}}\int_0^{\vartheta_{\max}} \frac{d\vartheta}{\sqrt{\left(\cos\vartheta - \cos\vartheta_{\max}\right)}}$$

$$T = 2\pi\sqrt{\frac{L}{g}}\left(1 + \frac{1}{16}\vartheta_{\max}^2 + \frac{11}{3072}\vartheta_{\max}^4 + \ldots\right)$$

$\theta_{MAX} = 5^0$	T= 1.00
$\theta_{MAX} = 30^0$	T= 1.02
$\theta_{MAX} = 60^0$	T= 1.07
$\theta_{MAX} = 90^0$	T= 1.18
$\theta_{MAX} = 150^0$	T= 1.76

L’ orbite





Orbite elliptique avec le soleil dans un foyer déjà imaginé par Hypatie d’Alexandrie ~400

a – axe majeur de l’ellipse

excentricité $e = \sqrt{1 - \frac{b^2}{a^2}}$

périhélie $q = (1 - e)\, a$ (plus proche)

aphélie $Q = (1 + e)\, a$ (plus éloigné)

Loi du mouvement (Loi de Newton)

$$\mathbf{F} = -\frac{GMm}{|\mathbf{r}|^3}\mathbf{r}$$

équations du mouvement

$$\begin{cases} m\dfrac{dv_x}{dt} = -GMm\dfrac{x}{r^3} \\ m\dfrac{dv_y}{dt} = -GMm\dfrac{y}{r^3} \end{cases}$$

$$r = \sqrt{x^2 + y^2}$$

en cordonnées polaires

$$r(\vartheta) = \frac{a\left(1 - e^2\right)}{1 - e\cos\vartheta}$$

* Équations de l'orbite (rappel des formules)

énergie : $E = T + U = \frac{1}{2} m v^2 - \frac{GMm}{r} < 0$

orbite circulaire : $E = -\frac{GMm}{2r}$ $\qquad v = \sqrt{\frac{GM}{r}}$

orbite elliptique : $E = -\frac{GMm}{2a}$ $\qquad v = \sqrt{GM\left(\frac{2}{r} - \frac{1}{a}\right)}$ a – axe majeur

3ème Loi de Kepler : $T^2 = \frac{4\pi^2}{GM} a^3$

quelques données
(Mercure, La Terre,
Jupiter, Uranus,
et la comète de Halley)

AU = unité astronomique ;
distance moyenne
Terre – Soleil ≈ 150 × 10^9 m

Nom	T (années)	ε excentricité	demi grand axe (AU)	*I* inclin.
Mercure	0.24	0.206	0.39	7^0
La Terre	1.00	0.017	1	0^0
Jupiter	11.86	0.05	5.20	1.31^0
Uranus	84.02	0.05	19.19	0.77^0
Halley	76.09	0.967	17.9	162.2^0

Rappel résolution numérique

À un moment donné t (conditions initiales), la planète est située en (x, y) avec la vitesse (v_x, v_y). Quelles sont la vitesse et la position au temps t + Δt ?

$$\mathbf{F} = -\frac{GMm}{|\mathbf{r}|^3}\mathbf{r}$$

Equations de Newton

$$m\frac{dv_x}{dt} = -GMm\frac{x}{r^3}$$

$$m\frac{dv_y}{dt} = -GMm\frac{y}{r^3}$$

$$r = \sqrt{x^2 + y^2}$$

système de 4 équations différentielles de premier ordre

$$\begin{cases} \frac{dv_x}{dt} = -GM\frac{x}{r^3} \\ \frac{dx}{dt} = v_x \\ \frac{dv_y}{dt} = -GM\frac{y}{r^3} \\ \frac{dy}{dt} = v_y \end{cases}$$

discrétisation du problème : e.g. $v_x(t) = \frac{dx}{dt} \rightarrow \frac{x(t+\Delta t) - x(t)}{\Delta t}$

A chaque instant t, la position et la vitesse au temps t + Δt sont (si Δt suffisamment petit) :

(Δt ~ 0.1% - 1% T)

$$x(t+\Delta t) = x(t) + v_x(t)\cdot\Delta t$$

$$v_x(t+\Delta t) = v_x(t) - GM\frac{x(t)}{\left(x^2(t)+y^2(t)\right)^{3/2}}\cdot\Delta t$$

même chose pour y …

1ère tentative

Calculez l'orbite, l'énergie (E, T, U) et le moment cinétique pour une comète par la méthode d'Euler pour une révolution (GM = 1)

a) sphérique: r(0) = 1 AU, vitesse tangentielle v(0) = 2π / an, Δt = 0.01 ans

b) elliptique: r(0) = 0.5 AU, y(0) = 0; $v_x(0) = 0$, $v_y(1.630)$, Δt = 0.1

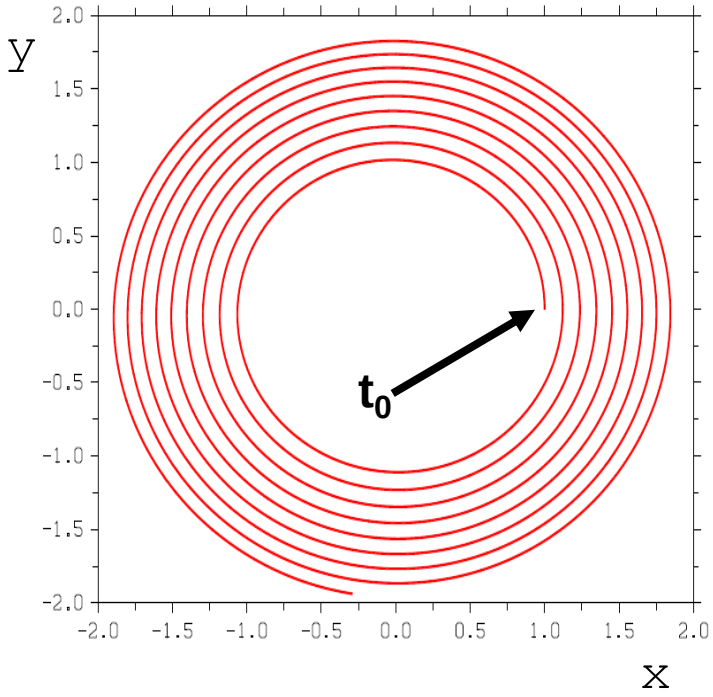
c) elliptique: r(0) = 1 AU; v(0) = π / an, Δt = 0.02 ans et Δt = 0.002 ans

Vous trouverez quelque chose comme ici à gauche.
L'orbite n'est pas fermée

Le problème est dû aux approximations faîtes pendant l'intégration de l'équation de l'orbite ;
les erreurs se cumulent toujours avec le même signe, donc l'orbite diverge.

Pour améliorer le résultat nous utiliserons la méthode de Runge.

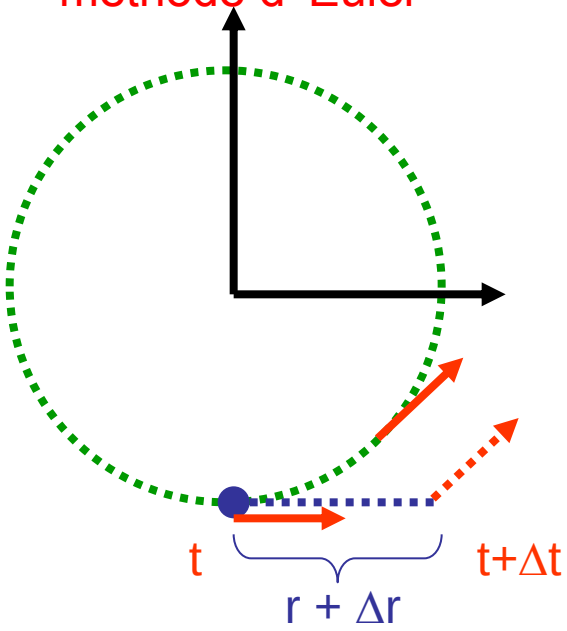
Solution d'Euler



voire Orbite_Euler.cpp

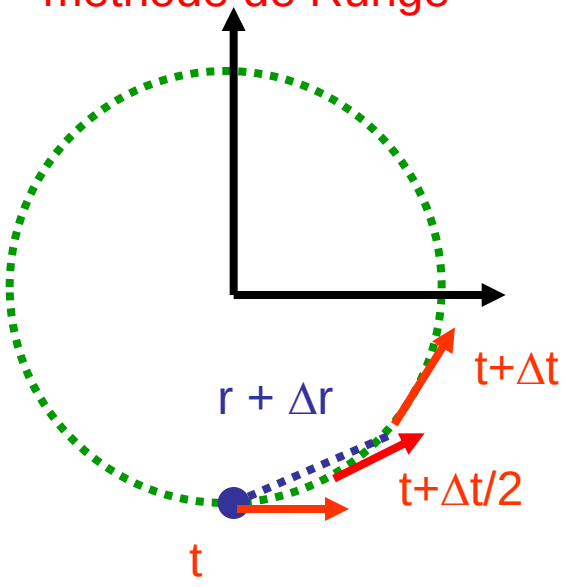
Où est le problème ?

méthode d' Euler



position → vitesse

méthode de Runge



vitesse → position

conditions initiales

$r(t_0) \quad v(t_0)$

$$r(t+\Delta t) = r(t) + v(t) \times \Delta t$$

$$v(t+\Delta t) = v(t) + a(r(t)) \times \Delta t$$

ancienne position et vitesse
⇒ nouvelle position et vitesse

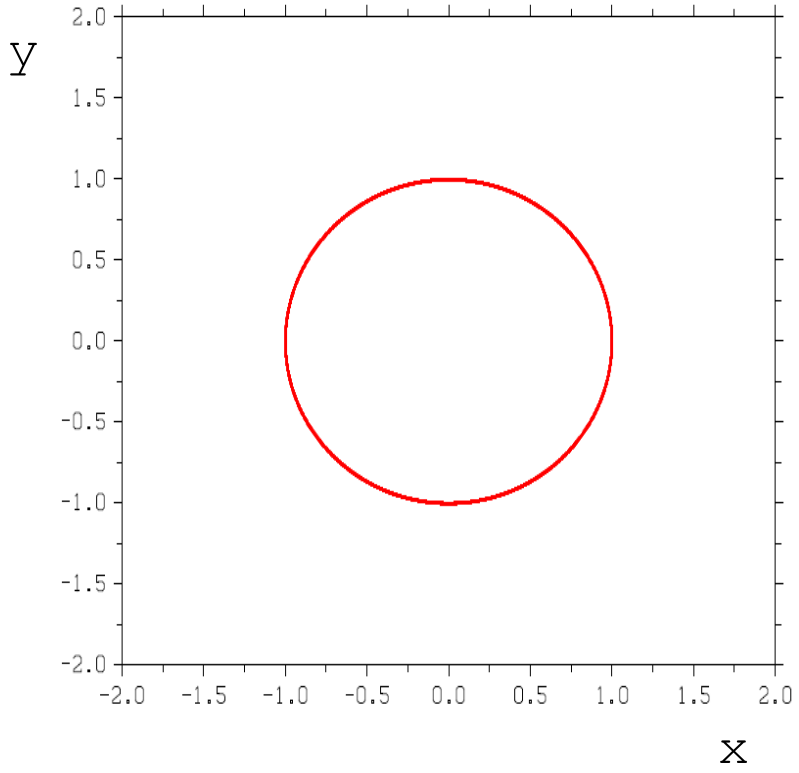
$$v(t_0) \rightarrow v(t_0 - \tfrac{1}{2}\Delta t) = v(t_0) - a(r(t_0)) \times \tfrac{1}{2}\Delta t \qquad r(t_0)$$

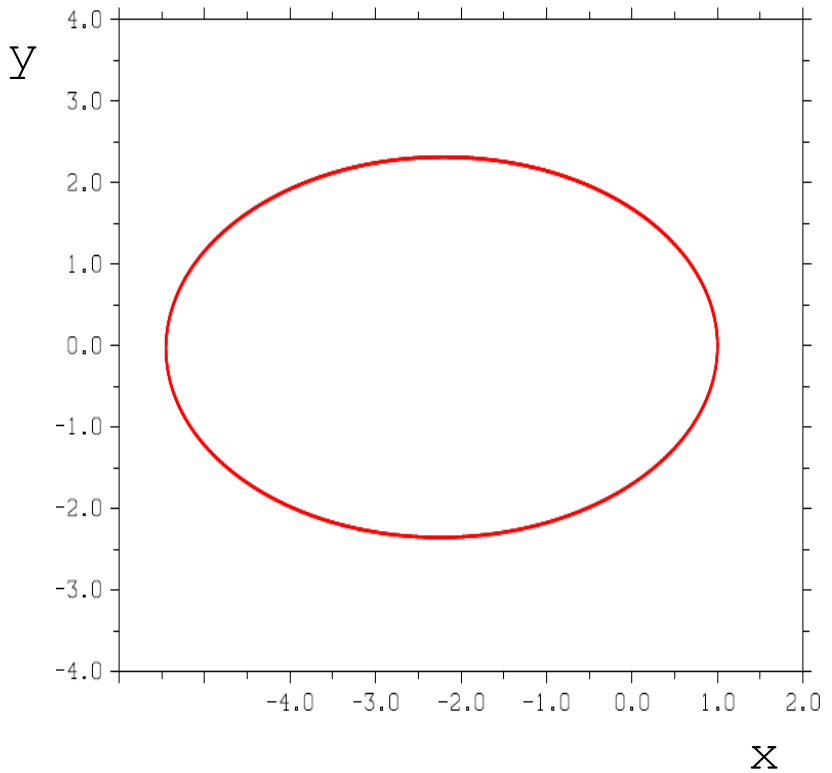
$$v\left(t + \tfrac{1}{2}\Delta t\right) = v(t - \tfrac{1}{2}\Delta t) + a(r(t)) \times \Delta t$$

$$r(t+\Delta t) = r(t) + v\left(t + \tfrac{1}{2}\Delta t\right) \times \Delta t$$

ancienne vitesse et position
⇒ nouvelle vitesse ⇒ nouvelle position

Solution de Runge





voire Orbite_Runge.cpp

Solution de Runge : l'orbite est fermée et l'énergie est conservée

La forme de l'orbite dépend des conditions initiales :
- l'orbite circulaire est un cas particulier,
- si la vitesse est trop élevée ($v > v_{fuite}$), la trajectoire sera une hyperbole,
- si la vitesse est trop petite, la planète tombera sur le Soleil.

La trajectoire n'est jamais une spirale !

Solution numérique

Equations de Newton

$$\mathbf{F} = -\frac{GMm}{|\mathbf{r}|^3}\mathbf{r}$$

$$m\frac{dv_x}{dt} = -GMm\frac{x}{r^3}$$

$$m\frac{dv_y}{dt} = -GMm\frac{y}{r^3}$$

$$r = \sqrt{x^2 + y^2}$$

système de 4 équations différentielles de premier ordre

$$\frac{dv_x}{dt} = -GM\frac{x}{r^3}$$

$$\frac{dx}{dt} = v_x$$

$$\frac{dv_y}{dt} = -GM\frac{y}{r^3}$$

$$\frac{dy}{dt} = v_y$$

algorithme méthode d'Euler

$$x[i+1] = x[i] + v_x[i]\times\Delta t$$

$$v_x[i+1] = v_x[i] - \frac{x[i]}{r^3}\times\Delta t$$

$$y[i+1] = y[i] + v_y[i]\times\Delta t$$

$$v_y[i+1] = v_y[i] - \frac{y[i]}{r^3}\times\Delta t$$

$$r = \sqrt{x[i]^2 + y[i]^2}$$

nouvelles conditions initiales

$$v_x[0] = v_x[0] + \frac{x[0]}{r^3}\times\frac{\Delta t}{2}$$

$$v_y[0] = v_y[0] + \frac{y[0]}{r^3}\times\frac{\Delta t}{2}$$

$$r = \sqrt{x[0]^2 + y[0]^2}$$

algorithme méthode de Runge

$$v_x[i+1] = v_x[i] - \frac{x[i]}{r^3}\times\Delta t$$

$$x[i+1] = x[i] + v_x[i+1]\times\Delta t$$

$$v_y[i+1] = v_y[i] - \frac{y[i]}{r^3}\times\Delta t$$

$$y[i+1] = y[i] + v_y[i+1]\Delta t$$

$$r = \sqrt{x[i]^2 + y[i]^2}$$

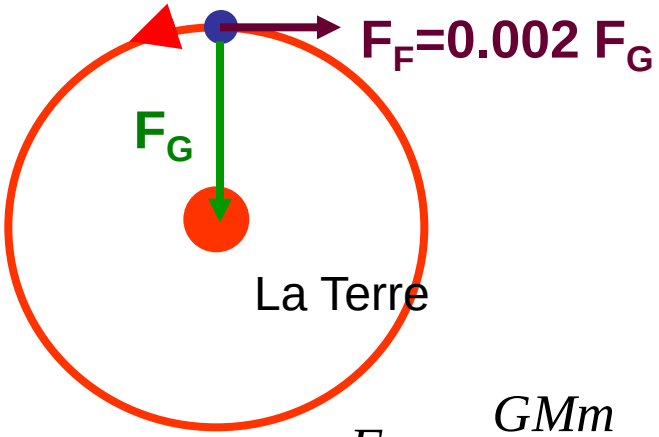
Avec frottement …

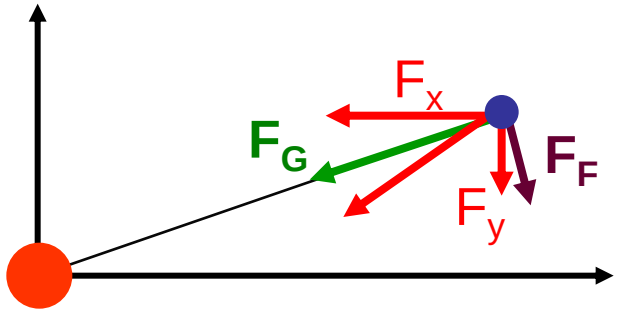
Un satellite est sur une orbite circulaire proche de la Terre, à 1000 km de la surface.
La force de frottement due à l'atmosphère représente 0.2% de la force gravitationnelle.
La force de frottement est toujours parallèle à la vitesse instantanée.
Estimez le temps qu'il faut pour que le satellite tombe sur la Terre.

Paramètres : $G_N = 6.673 \times 10^{-20}\ km^3\ kg^{-1}\ s^{-2}$,
$M_T = 5.974 \times 10^{24}\ kg$,
$R_T = 6.378 \times 10^{3}\ km$.

Le satellite est sur un orbite circulaire, donc sa vitesse est donnée par $v = \sqrt{\frac{G_N M_T}{R}}$

Conditions initiales : x(0) = R, y(0) = 0, $v_x(0) = 0$, $v_y(0) = v$. Utilisez $\Delta t = 1$ s.





$$F_x = -\frac{GMm}{r^2}\cos(\vartheta) + 0.002 \times \frac{GMm}{r^2}\sin(\vartheta) = -\frac{GMm}{r^2}\left(\frac{x}{r} - 0.002 \times \frac{y}{r}\right)$$

$$F_y = -\frac{GMm}{r^2}\sin(\vartheta) - 0.002 \times \frac{GMm}{r^2}\cos(\vartheta) = -\frac{GMm}{r^2}\left(\frac{y}{r} + 0.002 \times \frac{x}{r}\right)$$

voire Orbite3.cpp
et Orbite3Anima.cpp

Et si la force n'est pas $\propto 1/r^2$?

Loi gravitationnelle

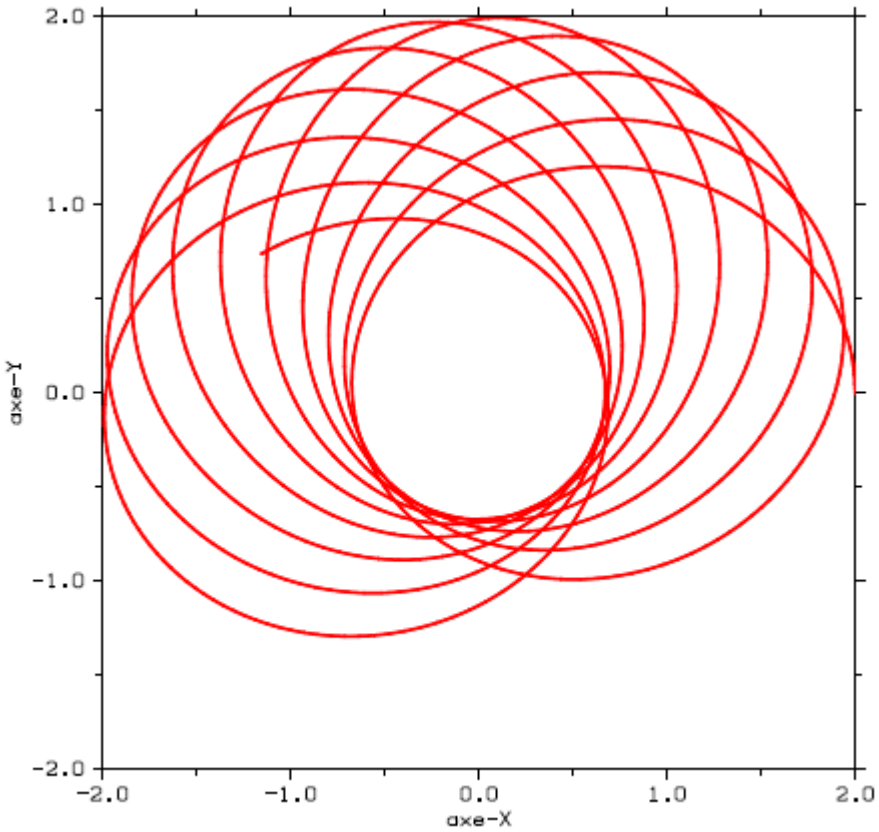
$$\mathbf{F}_G = -\frac{GMm}{|\mathbf{r}|^{\beta}}\frac{\mathbf{r}}{|\mathbf{r}|}$$

avec $\beta = 2$
(même chose pour la loi de Coulomb)

$\Rightarrow$ **L** est conservé
l'orbite est stable
ne précède pas

si $\beta \neq 2$
$\Rightarrow$ l'orbite n'est plus stable (pas fermé)
la planète tourne toujours autour du soleil
précession de l'orbite
bon test de la loi gravitationnelle

$\beta = 2.1$



voire Orbite4.cpp et Orbite4Anima.cpp

précession de l'orbite de Mercure: 5.66 arcsec / an (précession complète en 230000 an)
presque totalement expliqué par l'effet de Jupiter *et al.* (5.23 arcsec / an)
la petite différance (0.43 arcsec / an)
expliqué par la relativité générale

$$\mathbf{F}_G = -\frac{GMm}{|\mathbf{r}|^2}\frac{\mathbf{r}}{|\mathbf{r}|} \rightarrow\sim -\frac{GMm}{r^2}\left(1+\frac{1.1\times10^{-8}\,\mathrm{AU}^2}{r^2}\right)\frac{\mathbf{r}}{|\mathbf{r}|}$$

Classe `Planete`

Pour décrire le Système Solaire (plusieurs planètes), on procède de la même façon, comme pour la Terre, mais on se retrouve à dupliquer plusieurs fois les mêmes lignes de code. Mieux définir un nouveau type de données pour décrire les planètes et leur Mouvements $\Rightarrow$ classe `Planete`

chaque nouvelle planète sera un nouvel objet de cette classe

données pour décrire la planète (données membres de la classe `Planete`):

- nom
- rayon de l'orbite
- période de l'orbite
- excentricité de l'orbite
- inclination de l'orbite

et des données additionnelles calculées à partir de celles-ci:

- vitesse initiale
- position initiale

calculez avec la fonction `init` (méthode de la classe `Planete`)

pour initialiser l'objet on utilise les constructeurs
et une fonction `input` (méthode de la classe) pour saisir les données

et enfin pour calculer l'orbite la fonction `calcOrbite`

pour dessiner l'orbite, comme d'habitude, on utilisera la bibliothèque DISLIN.

```
class Planete {
  public:
    //constructeurs
    Planete();      //par defaut
    Planete(char *name, double r, double t, double ecc);
    //destructeur
    ~Planete();

    void input();     //saisie
    void calcOrbite(double t, double dt);     //calcule de l'orbite
    void position() const;      //affichage de la position de la planete

  private:
    void init();           //initialisation

    char m_nom[20];     //nom
    double m_a;      //axe major
    double m_T;      //periode
    double m_e;      //eccentricite
    double m_velIni;
    double m_posIni;
    double m_GM;      //G_N * M_Soleil

    double m_x0, m_y0;      //ancienne position
    double m_x, m_y;      // position actuell
    double m_t;      //temps
};
```

class Planete

constructeurs

méthodes publiques

méthode privée

données membres privées

voire Planete.cpp
et projet Planete

```
int main() {
  //creation des objets Planete
  Planete terre("Terre", 149.0, 365.26, 0.02);
  Planete venus("Venus", 108.2, 224.7, 0.01);
  Planete mars;
  mars.input();
  ....................

  terre.calcOrbite(time,dT);
  terre.position();
  venus.calcOrbite(time,dT);
  venus.position();
  mars.calcOrbite(time,dT);
  mars.position();
  ......................

  //DISLIN
  ............

  return 0;
}
```

initialisation lors de la déclaration

initialisation séparé

calcule de la position de la planète dans l'instant t_0 + time

affichage de la position

tous les paramètres relatifs à la planète sont enregistrés dans l'objet :

1. on n'a pas besoin de passer ces paramètre à la fonction «calcul de l'orbite»
2. ces paramètres ne sont pas visible, c.-à-d. il sont masqués (encapsulés)

Exercices – série 10

Problèmes

1. Etudiez les oscillations du pendule (p. 8 et 9)

2. Comparez les méthodes d'Euler et de Runge pour la résolution du mouvement du pendule.

3. Etudiez les oscillations d'un oscillateur anharmonique :
Pour α = 1 on retrouve l'oscillateur harmonique.

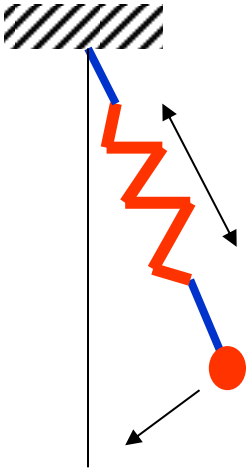
$$m\frac{d^2x}{dt^2} = -k \cdot x^{\alpha}$$

4. Un pendule physique est un pendule amorti et entretenu. L'équation du mouvement est donne par

$$\frac{d^2\vartheta(t)}{dt^2} = -\frac{g}{L}\sin\vartheta(t) - \gamma\frac{d\vartheta(t)}{dt} + F\sin(\Omega t)$$

ou γ est le coefficient d'amortissement et F la force d'entrainement de fréquence Ω. Etudiez le pendule avec la méthode de Runge.

5. Si la corde du pendule est élastique (ressort) calculez et dessinez la trajectoire du pendule (soit k la constante d'élasticité et L la longueur ; M la masse de la balle attachée à la corde).



6. Etudiez une orbite elliptique : r(0) = 1 AU et v(0) = 1.3 AU / an. Montrez que l'énergie cinétique et potentielle varient pendant la révolution mais que leur somme est constante. Montrez aussi que le moment cinétique est constant (i.e. dessinez E, T, U, L, …).
(voir Orbite_Euler.cpp et Orbite_Runge.cpp)

7. Un satellite est sur une orbite circulaire proche de la Terre, a 100 km de la surface. La force de frottement due a l'atmosphère représente 0.01% de la force gravitationnelle (situation hypothétique). La force de frottement est toujours parallèle à la vitesse instantanée. Estimez le temps qui il faut pour que le satellite descend à 50 km.
(voir Orbite3.cpp)

8. Une particule de masse M et charge Q se déplace dans un champ électrique **E** = (0, E_y, 0) et magnétique **B** = (0, 0, B_z). La particule est à repos dans le point **P** = (0, 0, 0) quand le champ électrique est allumé. Etudiez la trajectoire de la particule et dessinez la. Utilisez la méthode de Runge pour intégrer l'équation du mouvement. Quelle est la vélocité de dérive de la particule ? et dans quelle direction ? Estimez la à partir de la trajectoire.
Paramètres : M = 0.001 kg, Q = 0.001 C, E = 100 V, B = 1 T.

9. Etudiez un système des étoiles binaire pour deux étoiles de la même masse. Il vous faut écrire 4 équations différentielles de premier ordre pour chaque objet. Attention au conditions initiales.

10. Etudiez des orbites avec des classes (voir projet `planete`) :

class `Planete` – données de la planète

Suivez une procédure similaire à celle de la classe `Ressort` pour construire la classe `Planete`. Ignorez l'effet des autres planètes sur la planète étudiée.

Les données des planètes sont résumées ci-dessous.

Orbital and Historical Data

Name	#	Orbits	Distance (000 km)	O_Period (days)	Incl	Eccen	Discoverer	Date	A.
Sun	-	-	-	-	-	-	-	-	Sol
Mercury	I	Sun	57910	87.97	7.00	0.21	-	-	(0)
Venus	II	Sun	108200	224.70	3.39	0.01	-	-	(0)
Earth	III	Sun	149600	365.26	0.00	0.02	-	-	(0)
Mars	IV	Sun	227940	686.98	1.85	0.09	-	-	(0)
Jupiter	V	Sun	778330	4332.71	1.31	0.05	-	-	(0)
Saturn	VI	Sun	1429400	10759.50	2.49	0.06	-	-	(0)
Uranus	VII	Sun	2870990	30685.00	0.77	0.05	Herschel	1781	(0)
Neptune	VIII	Sun	4504300	60190.00	1.77	0.01	Adams(9)	1846	(0)
Pluto	IX	Sun	5913520	90550	17.15	0.25	Tombaugh	1930	(0)

Troisième Control Continu

28 Mai 2015
10h15 – 12h15

Salle 202 Science I

Amenez vos portables !

Vous pouvez utiliser toutes les notes du cours (incl. les corrigées 2013), des textes C++, …

interdit : e-mail, téléphone, des recherches sur la toile, skype, facebook …