I Résolution d'une équation différentielle d'ordre 1

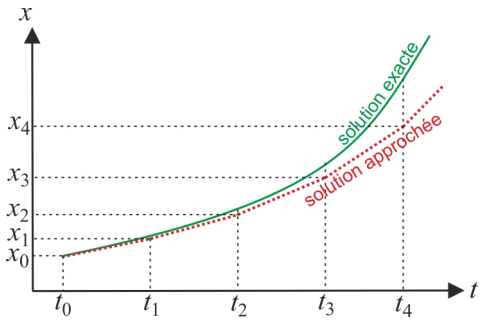
1. Méthode d'Euler

On cherche à déterminer la solution $x : \left|\begin{array}{ccc}\mathbf{R} & \longrightarrow & \mathbf{R}\\ t & \longmapsto & x(t)\end{array}\right.$ de l'équation différentielle $\dfrac{\mathrm{d}x}{\mathrm{d}t} = \varphi(t, x(t))$, linéaire ou pas, sur l'intervalle $[t_0, t_n]$, avec la condition initiale $x(t_0) = x_0$.

On divise l'intervalle $[t_0, t_n]$ en n intervalles $[t_i, t_{i+1}]$ $(i \in [\![0, n-1]\!])$ de longueur $\mathrm{d}t = \dfrac{t_n - t_0}{n}$ en posant $t_i = t_0 + i\mathrm{d}t$.
La solution **approchée** prend les valeurs x_i aux instants t_i.

La méthode (ou schéma) d'**Euler** consiste à approcher la dérivée $\dfrac{\mathrm{d}x}{\mathrm{d}t}(t_i)$ par la quantité $\dfrac{x_{i+1} - x_i}{\mathrm{d}t}$.

Cette approximation est bien sûr d'autant meilleure que l'intervalle de temps noté $\mathrm{d}t$ est petit. La solution obtenue serait exacte si $\mathrm{d}t$ était infiniment petit (d'ailleurs en Physique, $\mathrm{d}t$ désigne un infiniment petit).
La méthode d'Euler est **explicite** : la valeur approchée x_{i+1} de la solution à l'instant t_{i+1} se déduit de celle x_i à l'instant t_i par la relation $x_{i+1} = x_i + \varphi(t_i, x_i)\,\mathrm{d}t$, qui correspond à un développement limité à l'ordre 1 en $\mathrm{d}t$ (l'erreur commise à chaque itération est de l'ordre de $(\mathrm{d}t)^2$).
La courbe de la solution approchée est donc obtenue en partant du point $M_0(t_0, x_0)$ et en se déplaçant jusqu'au point $M_1(t_1, x_1)$ sur la tangente du graphe de f_0, solution de problème de Cauchy $\left\{\begin{array}{l}\dfrac{\mathrm{d}x}{\mathrm{d}t} = \varphi(t, x(t))\\ x(t_0) = x_0\end{array}\right.$, donc de la solution exacte.
Par la suite on se déplace de la même façon du point $M_i(t_i, x_i)$ jusqu'au point $M_{i+1}(t_{i+1}, x_{i+1})$ sur la tangente du graphe de f_i solution de problème de Cauchy $\left\{\begin{array}{l}\dfrac{\mathrm{d}x}{\mathrm{d}t} = \varphi(t, x(t))\\ x(t_i) = x_i\end{array}\right.$.
On représente ci-dessous le graphe de la solution exacte et de la solution approchée :



On peut diminuer l'écart entre la solution approchée et la solution exacte en diminuant la valeur de $\mathrm{d}t$.

On montre qu'il existe une constante C telle que, si x est la solution exacte du problème, $\forall i \in [\![0, n]\!], |x(t_i) - x_i| \leqslant C\mathrm{d}t$.
L'erreur globale commise est donc de l'ordre de $\mathrm{d}t$.

On constate les défauts de la méthode : les erreurs se cumulent et la solution approchée diverge rapidement de la solution exacte si $\mathrm{d}t$ n'est pas assez petit. La constante C dépend de la nature de la solution exacte.
La méthode d'Euler nécessite généralement de grandes valeurs de n pour obtenir une solution approchée satisfaisante. On dit qu'elle est facilement instable.

Prenons l'exemple de l'équation suivante : $\dfrac{\mathrm{d}x}{\mathrm{d}t} = \dfrac{1}{\tau}\left[-x + A\cos(\omega_0 t)\right]$, avec $\tau = 1,5$, $A = 1$ et $\omega_0 = 2\pi$. On étudie la solution entre $t_0 = 0$ et $t_n = 10$, avec pour condition initiale $x_0 = 1$. On prend d'abord $n = 1000$ intervalles de temps.
C'est l'équation qui régit un circuit passe-bas du premier ordre soumis à une excitation sinusoïdale.
Les calculs sont faits sous numpy et les graphes sous pyplot. Les valeurs numériques sont placées en variables globales :

```
import numpy as np
from matplotlib import pyplot as plt

t0,tn = 0.,10. # temps initial, temps final
x0 = 1 # valeur initiale de la fonction
tau,A,omega0 = 1.5,1,2*np.pi # valeur des paramètres
```

On commence par coder la fonction `phi(t,x)` qui renvoie $\dfrac{\mathrm{d}x}{\mathrm{d}t}$ à l'instant t :

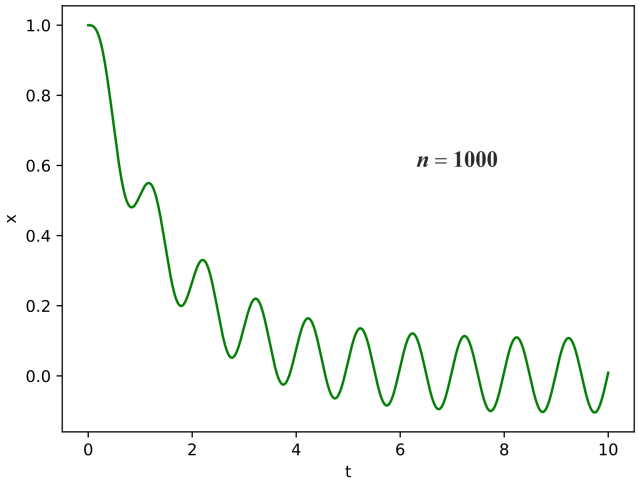
```
def phi(t,x) : # calcul de la dérivée de x
    xp = (-x+A*np.cos(omega0*t))/tau # calcul de dx/dt
    return xp # on renvoie dx/dt
```

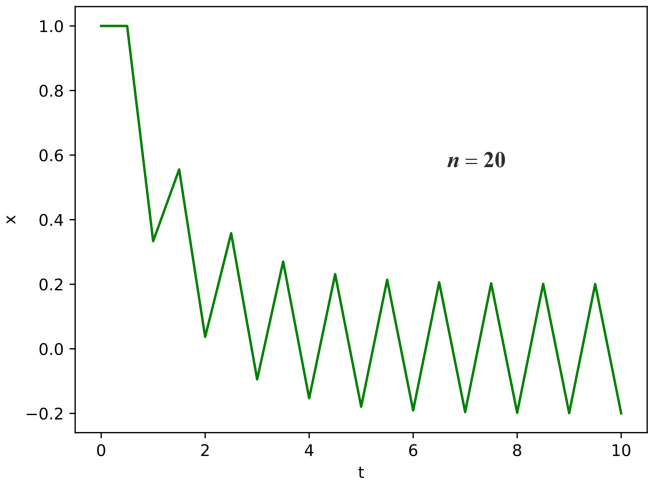
La fonction `Euler(phi,t0,tn,x0,n)` renvoie les vecteurs `T` et `X` qui contiennent respectivement les valeurs t_i et x_i pour $i \in [\![0, n-1]\!]$, calculées grâce aux relations de récurrence $t_{i+1} = t_i + \mathrm{d}t$ et $x_{i+1} = x_i + \varphi(t_i, x_i)\,\mathrm{d}t$:

```
def Euler(phi,t0,tn,x0,n): # calcul du vecteur T des valeurs de ti et X des valeurs de
    xi
    T = np.zeros(n+1) # initialisation de T avec des 0
    T[0] = t0
    X = np.zeros(n+1) # initialisation de X avec des 0
    X[0] = x0 # conditions initiales
    dt = (tn-t0)/n # pas
    for i in range(n): # méthode d'Euler
        T[i+1] = T[i]+dt
        X[i+1] = X[i]+phi(T[i],X[i])*dt
    return (T,X) # on renvoie T et X
```

Il ne reste plus qu'à tracer les solutions :

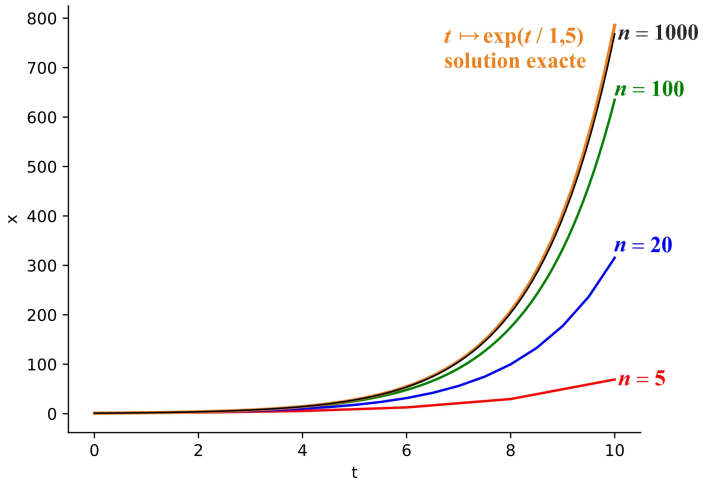
```
T,X = Euler(phi,t0,tn,x0,1000) #calcul des différentes solutions

#tracé des courbes
plt.figure("Solution")
plt.xlabel('t')
plt.ylabel('x')
plt.plot(T,X,color='green') # tracé de la solution x(t)
plt.show()
```



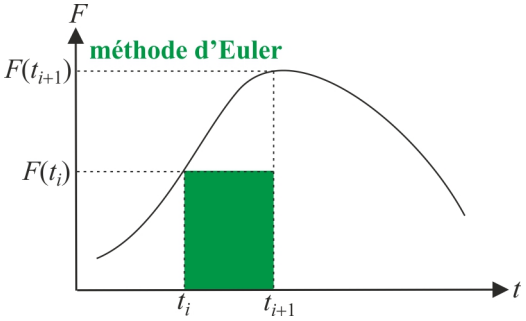


Le résultat est conforme aux prévisions : la solution de l'équation différentielle linéaire est la somme d'une solution particulière en $\cos(\omega_0 t + \psi)$ (régime sinusoïdal forcé) de période $T = \dfrac{2\pi}{\omega_0} = 1$ et d'une solution de l'équation homogène en $e^{-\frac{t}{\tau}}$ qui devient négligeable au bout de la durée caractéristique $3\tau = 4,5$.
Le critère de convergence n'est pas trop exigeant dans ce cas. Avec $n = 20$ le résultat reste proche de la solution exacte même si on ne reconnaît pas vraiment le régime sinusoïdal forcé car la résolution temporelle n'est pas suffisante.

Si on résout maintenant $\dfrac{\mathrm{d}x}{\mathrm{d}t} = \dfrac{x}{\tau}$ pour $\tau = 1,5$ sur l'intervalle $[t_0 = 0, t_n = 10]$ avec pour condition initiale $x(t_0) = 1$, on constate qu'il faut cette fois ci $n = 1000$ pour bien approcher la solution exacte.

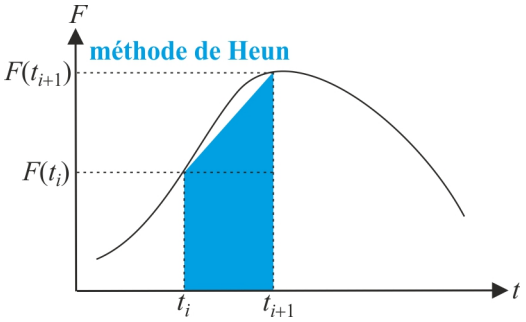


Avant d'explorer d'autres méthodes, remarquons que si l'on note F la fonction : $\left|\begin{array}{ccc}\mathbf{R} & \longrightarrow & \mathbf{R} \\ t & \longmapsto & F(t) = \varphi\left(t, x(t)\right)\end{array}\right.$, on peut écrire **sans** approximation la relation entre x_{i+1} et x_i :
$x_{i+1} = x_i + \displaystyle\int_{t_i}^{t_{i+1}} F(t)\mathrm{d}t$. L'intégrale intervenant est l'aire sous la courbe représentative de la fonction F entre t_i et t_{i+1}.
L'approximation d'Euler consiste à considérer que F est constante, égale à $F(t_i)$ entre t_i et t_{i+1}. C'est donc la méthode des rectangles :



2. Méthode de Heun

Dans la méthode de Heun, on approxime l'aire précédente par un trapèze : $\displaystyle\int_{t_i}^{t_{i+1}} F(t)\mathrm{d}t \approx \frac{1}{2}[F(t_i) + F(t_{i+1})]\mathrm{d}t$.



On obtient ainsi $x_{i+1} = x_i + \dfrac{1}{2}[\varphi(t_i, x_i) + \underbrace{\varphi(t_{i+1}, x_{i+1})}_{\text{inconnu}}]\mathrm{d}t$ (*).
La formule obtenue est **implicite** car le second membre dépend de x_{i+1}. On pourrait résoudre numériquement cette équation (par la méthode de Newton, ou par dichotomie) et la méthode serait qualifiée d'implicite. On peut éviter cette difficulté en utilisant la méthode d'Euler explicite pour calculer x_{i+1} au second membre : $\varphi(t_{i+1}, x_{i+1}) \approx \varphi(t_{i+1}, x_i + \varphi\left(t_i, x_i\right)\mathrm{d}t)$.
On parle alors de méthode **prédicteur-correcteur** car on utilise le schéma d'Euler pour prédire la valeur de x_{i+1}, puis on corrige en utilisant la formule (*).

On montre cette fois-ci qu'il existe une constante C telle que, si x est la solution exacte du problème, $\forall i \in [\![0, n]\!], |x(t_i) - x_i| \leqslant C(\mathrm{d}t)^2$. L'erreur globale commise est de l'ordre de $(\mathrm{d}t)^2$.

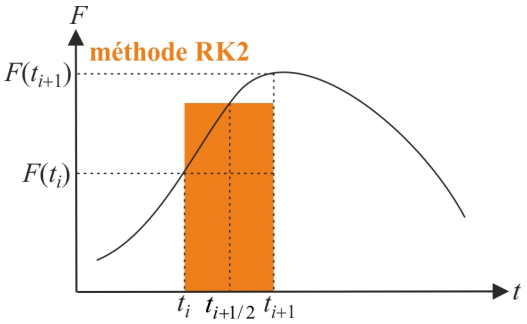
Le codage du schéma de Heun est similaire à celui du schéma d'Euler. On modifie uniquement la formule de récurrence de la fonction principale qui devient : $x_{i+1} = x_i + \frac{1}{2}[\varphi(t_i, x_i) + \varphi(t_{i+1}, x_i + \varphi(t_i, x_i)\,\mathrm{d}t)]\mathrm{d}t$:

```
def Heun(phi,t0,tn,x0,n): # calcul du vecteur T des valeurs de ti et X des valeurs de
    xi
    T = np.zeros(n+1) # initialisation de T avec des 0
    T[0] = t0
    X = np.zeros(n+1) # initialisation de X avec des 0
    X[0] = x0 # conditions initiales
    dt = (tn-t0)/n # pas
    for i in range(n): # méthode de Heun
        T[i+1] = T[i]+dt
        K = phi(T[i],X[i]) # on ne calcule cette grandeur qu'une seule fois
        X[i+1] = X[i]+0.5*(K+phi(T[i+1],X[i]+K*dt))*dt
    return (T,X) # on renvoie T et X
```

3. Méthode de Runge-Kutta d'ordre 4 (RK4)

L'idée est d'améliorer la précision en augmentant le nombre d'évaluations à chaque pas de temps. **RK1** (une seule évaluation de F) est donc la méthode d'Euler.

Pour **RK2**, on pose $t_{i+\frac{1}{2}} = \frac{1}{2}[t_i + t_{i+1}]$ et on fait l'approximation que F est constante, égale à $F(t_{i+\frac{1}{2}})$ entre t_i et t_{i+1}.



On a donc $x_{i+1} = x_i + \underbrace{\varphi(t_{i+\frac{1}{2}}, x_{i+\frac{1}{2}})}_{\text{inconnu}}\,\mathrm{d}t$ (*).

Comme précédemment, $x_{i+\frac{1}{2}}$ est prédit par le schéma d'Euler : $x_{i+\frac{1}{2}} = x_i + \varphi(t_i, x_i)\frac{\mathrm{d}t}{2}$ (schéma prédicteur-correcteur)

À chaque itération, on effectue les calculs suivants :

- calcul de $K_1 = \varphi(t_i, x_i)$
- calcul de la dérivée $\frac{\mathrm{d}x}{\mathrm{d}t}$ au milieu du pas : $K_2 = \varphi(t_{i+\frac{1}{2}}, x_{i+\frac{1}{2}}) = \varphi\left(t_i + \frac{\mathrm{d}t}{2}, x_i + K_1\frac{\mathrm{d}t}{2}\right)$
- calcul de x_{i+1} à l'aide de cette dérivée : $x_{i+1} = x_i + K_2\mathrm{d}t$.

On montre que l'erreur globale du schéma est de l'ordre de $(\mathrm{d}t)^2$.

Le schéma **RK4** est très utilisé car il l'erreur globale est de l'ordre de $(\mathrm{d}t)^4$. On montre en effet que pour ce schéma, il existe une constante C telle que, si x est la solution exacte du problème, $\forall i \in [\![0, n]\!]\,, |x(t_i) - x_i| \leqslant C(\mathrm{d}t)^4$.

À chaque itération, on effectue pour le schéma RK4 les calculs suivants :

- calcul de $K_1 = \varphi(t_i, x_i)$
- calcul de $K_2 = \varphi\left(t_i + \frac{\mathrm{d}t}{2}, x_i + K_1\frac{\mathrm{d}t}{2}\right)$
- calcul de $K_3 = \varphi\left(t_i + \frac{\mathrm{d}t}{2}, x_i + K_2\frac{\mathrm{d}t}{2}\right)$
- calcul de $K_4 = \varphi(t_i + \mathrm{d}t, x_i + K_3\mathrm{d}t)$
- calcul de $x_{i+1} = x_i + \frac{\mathrm{d}t}{6}[K_1 + 2K_2 + 2K_3 + K_4]$.

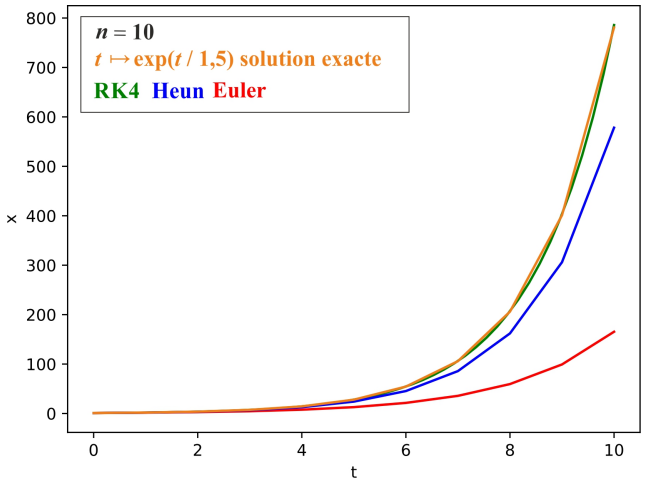
Ainsi, la pente est obtenue par une moyenne pondérée de pentes : K_1 est la pente au début de l'intervalle ; K_2 est la pente au milieu de l'intervalle, calculée en utilisant K_1 ; K_3 est la pente au milieu de l'intervalle, calculée en utilisant K_2 ; K_4 est la pente à la fin de l'intervalle, calculée en utilisant K_3. Dans la moyenne des quatre pentes, un poids plus grand est donné aux pentes au point milieu.

Pour le codage du schéma RK4 est similaire à celui du schéma d'Euler, on introduit ces 4 étapes dans la formule de récurrence de la fonction principale :

```
def RK4(phi,t0,tn,x0,n): # calcul du vecteur T des valeurs de ti et X des valeurs de xi
    T = np.zeros(n+1) # initialisation de T avec des 0
    T[0] = t0
    X = np.zeros(n+1) # initialisation de X avec des 0
    X[0] = x0 # conditions initiales
    dt = (tn-t0)/n # pas
    for i in range(n): # méthode RK4
        T[i+1] = T[i]+dt
        K1 = phi(T[i],X[i])
        K2 = phi(T[i]+.5*dt,X[i]+K1*.5*dt)
        K3 = phi(T[i]+.5*dt,X[i]+K2*.5*dt)
        K4 = phi(T[i]+dt,X[i]+K3*dt)
        X[i+1] = X[i]+dt/6*(K1+2*K2+2*K3+K4)
    return (T,X) # on renvoie T et X
```

4. Comparaison des méthodes et conclusion

Comparons les résultats obtenus avec les 3 schémas (Euler, Heun, RK4) dans le cas de l'équation différentielle $\dfrac{\mathrm{d}x}{\mathrm{d}t} = \dfrac{x}{\tau}$ avec $\tau = 1,5$, sur l'intervalle $[t_0 = 0, t_n = 10]$ et avec pour condition initiale $x(t_0) = 1$. La solution exacte est $t \longmapsto e^{t/1,5}$.
On prend à chaque fois le même faible nombre d'intervalles ($n = 10$) :



On vérifie que pour un intervalle temporel $\Delta t = t_n - t_0 = ndt$ fixé, le schéma d'Euler, de l'ordre de $\dfrac{1}{n}$, nécessite de grandes valeurs de n pour donner une solution approchée correcte. Avec la même valeur de n, le schéma de Heun, en $\dfrac{1}{n^2}$ donne une meilleure approximation, et le schéma RK4, en $\dfrac{1}{n^4}$, encore une meilleure approximation.
Le schéma d'Euler présente un fort intérêt pédagogique et est utilisé lors des étapes des schémas de Heun et de RK4, mais il nécessite de nombreuses itérations pour donner un résultat correct.
La "bonne valeur" du pas temporel utilisé (donc de n) résulte d'un compromis. Si n est trop petit, la solution obtenue n'est pas une bonne approximation de la solution cherchée. Si n est trop grand, le temps de calcul peut augmenter considérablement, et d'autre part, à chaque pas les grandeurs évaluées varient très peu si bien que les erreurs d'arrondi peuvent devenir prédominantes.

II Résolution d'une équation différentielle d'ordre 2

On cherche à déterminer la solution $x : \left| \begin{array}{ccc} \mathbf{R} & \longrightarrow & \mathbf{R} \\ t & \longmapsto & x(t) \end{array} \right.$ de l'équation différentielle $\dfrac{\mathrm{d}^2x}{\mathrm{d}t^2} = \varphi\left(t, x, \dfrac{\mathrm{d}x}{\mathrm{d}t}\right)$ linéaire ou pas avec les conditions initiales $x(t_0) = x_0$ et $\dfrac{\mathrm{d}x}{\mathrm{d}t}(t_0) = x_0'$ sur l'intervalle $[t_0, t_n]$.

On se ramène à une équation différentielle d'ordre 1 en posant $X = \begin{bmatrix} x \\ \dfrac{\mathrm{d}x}{\mathrm{d}t} \end{bmatrix}$.

La dérivée de ce vecteur est $\dfrac{\mathrm{d}X}{\mathrm{d}t} = \left[\begin{array}{l} \dfrac{\mathrm{d}x}{\mathrm{d}t} \\ \dfrac{\mathrm{d}^2x}{\mathrm{d}t^2} = \varphi\left(t, x, \dfrac{\mathrm{d}x}{\mathrm{d}t}\right) \end{array}\right]$.

On divise toujours l'intervalle $[t_0, t_n]$ en n intervalles $[t_i, t_{i+1}]$ $(0 \leqslant i \leqslant n-1)$ de longueur $\mathrm{d}t = \dfrac{t_n - t_0}{n}$ en posant $t_i = t_0 + i\mathrm{d}t$. Il ne reste plus qu'à définir la fonction Φ telle que $\dfrac{\mathrm{d}X}{\mathrm{d}t} = \Phi(t, X(t))$ pour appliquer les méthodes précédentes (Euler, Heun, RK4) comme pour des équations d'ordre 1.

Prenons l'exemple de l'équation suivante : $\boxed{\dfrac{\mathrm{d}^2x}{\mathrm{d}t^2} = \epsilon\omega_0(1-x^2)\dfrac{\mathrm{d}x}{\mathrm{d}t} - \omega_0^2 x}$ (équation de Van Der Pol), avec $\omega_0 = 2\pi$.

La fonction Φ qui renvoie $\dfrac{\mathrm{d}X}{\mathrm{d}t}$ se code ainsi :

```
def Phi(t,X) : # calcul de la dérivée de X=[x,dx/dt] à l'instant ti
    x,xp = X[0],X[1] # on extrait x(ti) et dx/dt(ti)
    xpp = epsilon*omega0*(1-x**2)*xp-omega0**2*x # calcul de d2x/dt2
    return np.array([xp,xpp]) # on renvoie le vecteur dX/dt(ti)=[dx/dt(ti),d2x/dt2(ti)]
        qui va permettre de calculer X(ti+1) par la méthode d'Euler
```

On étudie la solution entre $t_0 = 0$ et $t_n = 5$, avec pour conditions initiales $x_0 = 1$ et $x_0' = 0$. On prend $n = 500$ intervalles de temps.

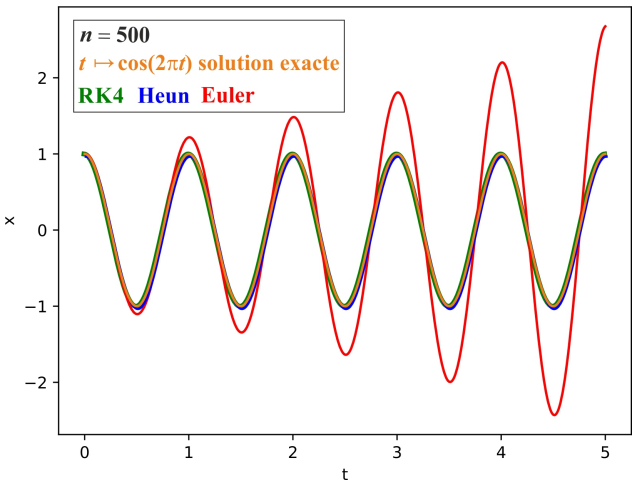
Nous commençons par le cas $\epsilon = 0$: on retrouve l'équation d'un oscillateur harmonique de période $T = \dfrac{2\pi}{\omega_0} = 1$. La solution exacte est ici $t \longmapsto \cos(\omega_0 t)$.

Voici le code permettant de tracer la solution approchée de l'équation différentielle. On utilise les trois méthodes précédentes. Les arguments sont `Phi`, les instants initiaux et finaux `t0` et `tn`, les conditions initiales `x0` et `xp0` et enfin `n`, le nombre de sous-intervalles :

```
def Euler(Phi,t0,tn,x0,xp0,n): # calcul du vecteur T des valeurs de ti et X tableau
   dont la ligne i est X(ti)=[x(ti),dx/dt(ti)]
    T = np.zeros(n+1) # initialisation de T avec des 0
    T[0] = t0
    X = np.zeros((n+1,2)) # initialisation de X avec des [0,0]
    X[0] = [x0,xp0] # conditions initiales
    dt = (tn-t0)/n # pas
    for i in range(n): # méthode d'Euler
        T[i+1] = T[i]+dt
        X[i+1] = X[i]+Phi(T[i],X[i])*dt
    return (T,X) # on renvoie T et X

def Heun(Phi,t0,tn,x0,xp0,n):
    T = np.zeros(n+1)
    T[0] = t0
    X = np.zeros((n+1,2))
    X[0] = [x0,xp0]
    dt = (tn-t0)/n
    for i in range(n): # méthode de Heun
        T[i+1] = T[i]+dt
        K = Phi(T[i],X[i]) # on ne calcule cette grandeur qu'une seule fois
        X[i+1] = X[i]+0.5*(K+Phi(T[i+1],X[i]+K*dt))*dt
    return (T,X)

def RK4(Phi,t0,tn,x0,xp0,n):
    T = np.zeros(n+1)
    T[0] = t0
    X = np.zeros((n+1,2))
    X[0] = [x0,xp0]
    dt = (tn-t0)/n
    for i in range(n): # méthode RK4
        T[i+1] = T[i]+dt
        K1 = Phi(T[i],X[i])
        K2 = Phi(T[i]+.5*dt,X[i]+K1*.5*dt)
        K3 = Phi(T[i]+.5*dt,X[i]+K2*.5*dt)
```

```
        K4 = Phi(T[i]+dt,X[i]+K3*dt)
        X[i+1] = X[i]+dt/6*(K1+2*K2+2*K3+K4)
    return (T,X)

t0,tn = 0,5. # temps initial, temps final
n = 500 # nombre d'intervalles
x0,xp0 = 1,0 # valeur initiale de la fonction et de sa dérivée
epsilon = 0 # valeur du premier paramètre
omega0 = 2*np.pi # valeur du second paramètre

def  fonction(t): # solution exacte
    return np.cos(omega0*t)
T1 = np.linspace(t0,tn,n)
x1 = fonction(T1)
T2,X = Euler(Phi,t0,tn,x0,xp0,n) #calcul des différentes solutions
x2 = X[:,0] # on extrait le vecteur contenant les x(ti)
T3,X = Heun(Phi,t0,tn,x0,xp0,n) #calcul des différentes solutions
x3 = X[:,0] # on extrait le vecteur contenant les x(ti)
T4,X = RK4(Phi,t0,tn,x0,xp0,n) #calcul des différentes solutions
x4 = X[:,0] # on extrait le vecteur contenant les x(ti)

#tracé des courbes
plt.xlabel('t')
plt.ylabel('x')
plt.plot(T1,x1,color='green') # tracé de la solution x(t)
plt.plot(T2,x2,color='red') # tracé de la solution x(t)
plt.plot(T3,x3,color='blue') # tracé de la solution x(t)
plt.plot(T4,x4,color='yellow') # tracé de la solution x(t)
plt.show()
```

On extrait le vecteur `x` donnant les valeurs $x(t_i)\ \forall i \in [\![0,n]\!]$ par slicing à partir du vecteur `X` grâce à la commande : `x = X[:,0]`.
Alors qu'avec 500 pas de temps les schémas de Heun et RK4 permettent d'approcher la solution exacte, on constate que la méthode d'Euler est instable et que la solution obtenue diverge rapidement, ce qui obligerait à prendre des valeurs de n beaucoup plus grandes que 500 pour pouvoir approcher $t \longmapsto \cos(\omega_0 t)$.



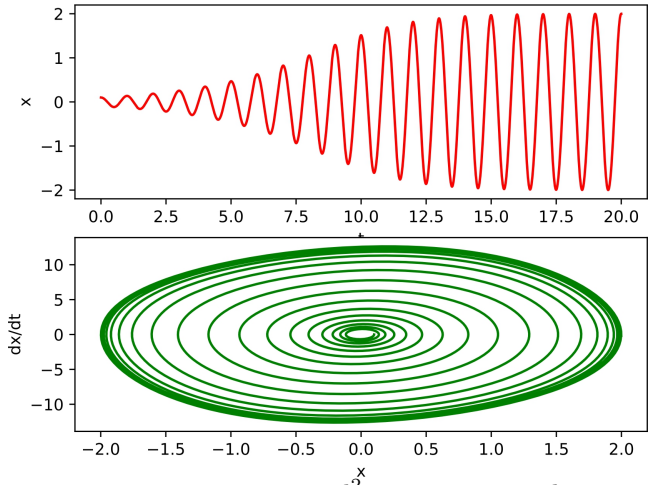
Pour finir, conservons le schéma RK4 pour observer le comportement d'un oscillateur de Van Der Pol, qui correspond à $\epsilon > 0$. Pour un système d'ordre 2, il est intéressant de tracer également la trajectoire de phase dans l'espace $\left(x, \dfrac{\mathrm{d}x}{\mathrm{d}t}\right)$. On obtient ce tracé en extrayant également le vecteur `xp` donnant les valeurs $x'(t_i)\ \forall i \in [\![0,n]\!]$ par slicing à partir du vecteur `X` grâce à la commande : `xp = X[:,1]`.

```
epsilon = 0.5 # valeur du premier paramètre
#tracé des courbes
T,X = RK4(Phi,t0,tn,x0,xp0,n) #calcul des différentes solutions
x = X[:,0] # on extrait le vecteur contenant les x(ti)
xp = X[:,1] # on extrait le vecteur contenant les dx/dt(ti)
```

```
plt.figure("Solution")
plt.subplot(2,1,1) #nbre de lignes, nbre de colonnes, n° de la figure du tableau de
    graphiques
plt.xlabel('t')
plt.ylabel('x')
plt.plot(T,x,color='red') # tracé de la solution x(t)
plt.subplot (2,1,2)
plt.xlabel('x')
plt.ylabel('dx/dt')
plt.plot(x,xp,color='green') # tracé du portrait de phase
plt.show()
```

On étudie la solution entre $t_0 = 0$ et $t_n = 20$, avec pour conditions initiales $x_0 = 0,1$ et $x'_0 = 0$. On prend $n = 5000$ intervalles de temps.

— Pour $\epsilon = 0,1$ on obtient les courbes suivantes :



L'analyse physique du phénomène est simple : l'équation $\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + \epsilon\omega_0(x^2-1)\frac{\mathrm{d}x}{\mathrm{d}t} + \omega_0^2 x = 0$ est celle d'un oscillateur harmonique avec un terme en $\frac{\mathrm{d}x}{\mathrm{d}t}$ correspondant à une perte d'énergie si $|x| > 1$ car il est alors de la forme $2\sigma\omega_0\frac{\mathrm{d}x}{\mathrm{d}t}$ avec $\sigma > 0$, et à un gain d'énergie si $|x| < 1$. L'oscillateur passe donc d'une phase d'amplification lorsque $|x| < 1$ à une phase d'amortissement quand $|x|$ devient supérieur à 1. Après un régime transitoire, on atteint un régime périodique plus ou moins sinusoïdal (ce que l'on voit mieux sur la trajectoire de phase, proche d'une ellipse si les oscillations sont quasi-sinusoïdales). On constate que plus ϵ est petit, plus le régime transitoire est long et plus x varie de façon sinusoïdale en régime établi.

— En revanche, comme on le voit ci-dessous pour $\epsilon = 0,5$, après un régime transitoire très court, la courbe représentative de $x(t)$ s'écarte fortement d'une sinusoïde.