CN – TP 2

Euler explicite – Circuit d'ordre un

I - Méthode d'Euler explicite

I.1 - Définition

La méthode d'Euler explicite est une **méthode itérative** qui permet de déterminer numériquement une **solution approchée** de l'équation différentielle

$$\boxed{y'(t) = F(y(t), t)}$$

sur l'intervalle $[t_0, t_f]$, avec la condition initiale $y(t_0) = y_0$ (problème de Cauchy).
La procédure algorithmique est la suivante :

— On commence par réaliser une subdivision régulière de l'intervalle $[t_0, t_f]$ en sous-intervalles de largeur δt (δt est le **pas** de résolution), ce qui revient à générer un ensemble de points d'abscisses t_k telles que $t_k = t_0 + k \times \delta t$ et $t_{k+1} - t_k = \delta t$.

— Par définition, on a

$$y'(t_k) = \lim_{\delta t \to 0} \frac{y(t_k + \delta t) - y(t_k)}{\delta t} = \lim_{(t_{k+1} - t_k) \to 0} \frac{y(t_{k+1}) - y(t_k)}{t_{k+1} - t_k}$$

et d'après l'équation différentielle qu'on cherche à résoudre, on a

$$\boxed{y'(t_k) = F(y(t_k), t_k)}$$

On va chercher les valeurs numériques approchées des $y(t_k)$; on note ces valeurs approchées y_k. On a donc

$$F(y(t_k), t_k) \approx F(y_k, t_k)$$

D'autre part, si le pas de résolution est suffisamment petit, on peut écrire avec une bonne approximation que

$$y'(t_k) \approx \frac{y_{k+1} - y_k}{t_{k+1} - t_k} = \frac{y_{k+1} - y_k}{\delta t}$$

Finalement, dans le cadre de l'approximation

$$\frac{y_{k+1} - y_k}{\delta t} = F(y_k, t_k)$$

y_0 étant connu grâce à la condition initiale, on détermine les valeurs y_k en procédant de proche en proche grâce à la **relation de récurrence** (ou **schéma numérique**) :

$$\boxed{y_{k+1} = y_k + F(y_k, t_k) \times \delta t}$$

L'approximation réalisée est d'autant meilleure que le pas de résolution est petit. Le choix du pas de résolution est toujours l'objet d'un compromis :

— trop petit, il mène à une bonne approximation mais à un nombre d'itérations de l'algorithme important et donc un temps de calcul important également ;

— trop grand, il mène à des temps de calcul faible mais à une mauvaise approximation.

On montre par la suite quelques exemples nous permettant de trouver un pas de résolution menant à une approximation et des temps de calcul acceptables, notamment en choisissant de façon raisonnable l'intervalle $[t_0, t_f]$ et le nombre de sous-intervalles.

I.2 - Exemple d'implémentation en langage Python

```python
def euler(F, y0, t0, tf, dt):
    """
    Résout le problème de Cauchy y'(t)=F(y(t),t) avec y(0)=y0 par la méthode
    d'Euler explicite.
    Arguments d'entrée :
        - F : fonction donnant y' (fonction de 2 variables) ;
        - y0 : condition initiale sur y (flottant) ;
        - t0 et tf : bornes de l'intervalle de résolution (flottants) ;
        - dt : pas de discrétisation utilisé pour la résolution (flottant).
    Variables de sortie :
        - t : vecteur contenant l'ensemble des instants tk (array numpy) ;
        - y : vecteur contenant l'ensemble des valeurs approchées yk (array numpy).
    """
    # Initialisation des variables de sortie
    t = np.arange(t0, tf+dt, dt)    # la dernière valeur est dans [tf,tf+dt[
    N = len(t)
    y = np.zeros(N)                 # initialisation de l'array y
    y[0] = y0                       # prise en compte de la CI

    # Boucle permettant le calcul des yk par récurrence
    for k in range(0,N-1):
        y[k+1] = ??????????????

    return t, y
```

II - Simulation de la réponse d'un circuit RC à un échelon de tension

Manipulation 1
Faire le schéma d'un circuit RC soumis à un échelon de tension E. Retrouver l'équation différentielle régissant la tension $u(t)$ aux bornes du condensateur. Résoudre cette équation différentielle pour en déduire la solution exacte $u_e(t)$, à partir de la conditions initiale $u(0) = U_0$.

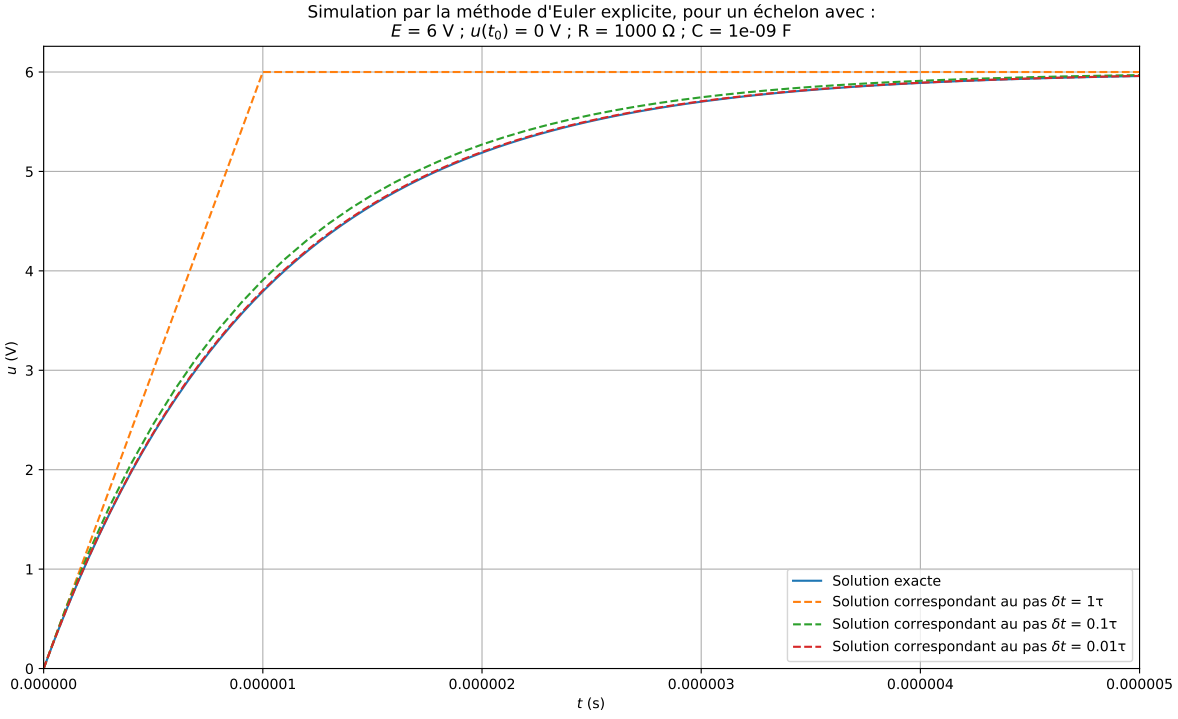
Pour l'évolution temporelle de la réponse d'un circuit RC, il parait logique de déterminer l'intervalle $[t_0, t_f]$ en fonction du temps de relaxation. En effet, on sait que la durée du régime transitoire dépend de ce temps de relaxation et qu'« au bout de quelques τ », le régime transitoire est approximativement terminé.

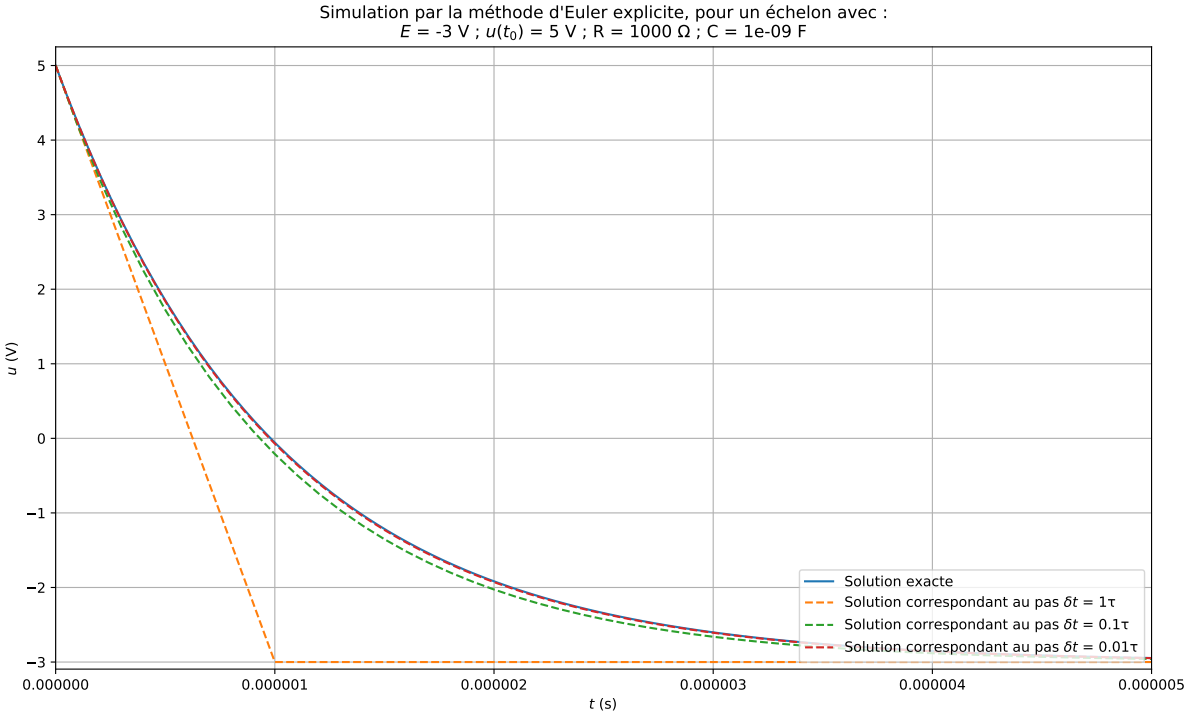
Manipulation 2
Récupérer le début de programme python disponible sur le site et dans le dossier Ressources (`CN-TP2-Euler-echelon-RC-start`) et compléter les lignes ci-dessous, de façon à simuler la réponse d'un circuit RC à un échelon de tension par la méthode d'Euler.

```python
def euler(F, y0, t0, tf, dt):
    """
    Résout le problème de Cauchy y'(t)=F(y(t),t) avec y(0)=y0 par la méthode
    d'Euler explicite.
    Arguments d'entrée :
        - F : fonction donnant y' (fonction de 2 variables y et t) ;
        - y0 : condition initiale sur y (flottant) ;
        - t0 et tf : bornes de l'intervalle de résolution (flottants) ;
        - dt : pas de discrétisation utilisé pour la résolution (flottant).
    Variables de sortie :
        - t : vecteur contenant l'ensemble des instants tk (array numpy) ;
        - y : vecteur contenant l'ensemble des valeurs approchées yk (array numpy).
    """
    # Initialisation des variables de sortie
```

```
    t = np.arange(t0, tf+dt, dt)    # la dernière valeur est dans [tf,tf+dt[
    N = len(t)
    y = np.zeros(N)                 # initialisation du array y
    y[0] = y0                       # prise en compte de la CI

    # Boucle permettant le calcul des yk par récurrence
    for k in range(0,N-1):
        y[k+1] = ??????????????????????

    return t, y # retourne un tableau contenant les temps tk
                # et un tableau contenant les valeurs approchées yk

def circuitRC(u, t):
    """
    Fonction explicitant du/dt en fonction de u et t.
    On a F(u(t),t) = du/dt = E/tau - u(t)/tau.
    """
    return ???????????

def u_exacte(t):
    """
    Expression analytique de la solution exacte (forme adimensionnée, correspondant à la
    condition initiale u(0) = u0).
    On a u_exacte(t) = ????????????????????
    """
    return ??????????????????
```

Une fois la largeur de l'intervalle choisie, il faut choisir un pas de résolution δt. On présente ci-après différentes simulations, comparées à la solution analytique exacte pour des pas de résolution $\delta t = \tau$, $\delta t = 0,1\tau$ et $\delta t = 0,01\tau$.





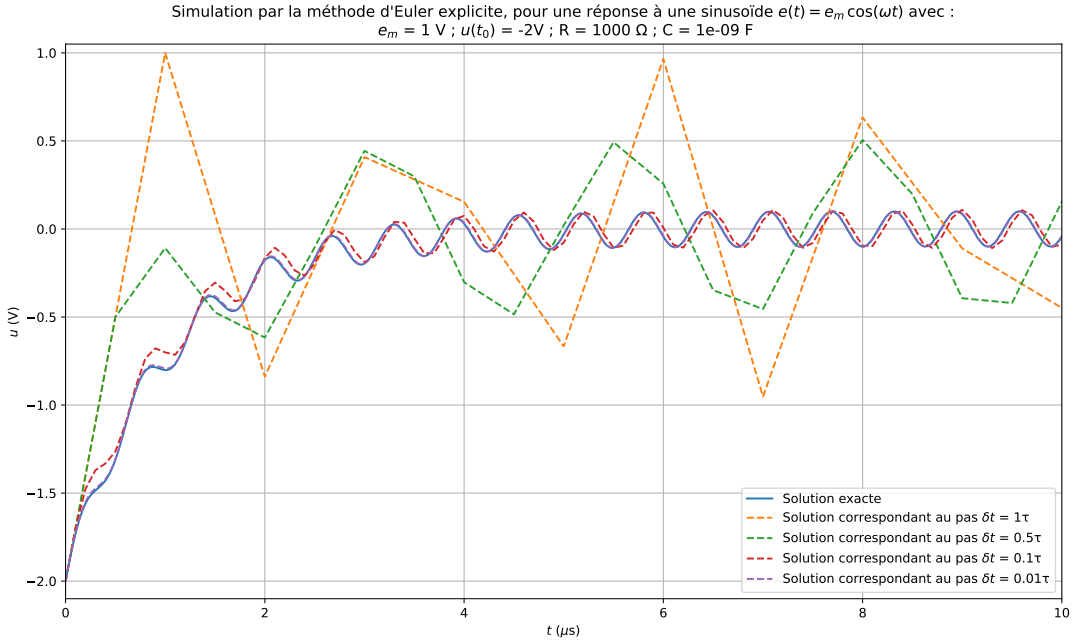
On constate qu'un pas de résolution δt compris entre $0,01\tau$ et $0,1\tau$ donne des résultats satisfaisants. La simulation avec un pas $\delta t = \tau$ n'est pas satisfaisante du tout mais, de façon intéressante, elle permet de visualiser la tangente à l'origine.

Manipulation 3
Compléter le code écrit précedemment pour reproduire les figures ci-dessus, en ajoutant un pas de résolution égal à $0,2\tau$.

III - Autres formes d'excitations

L'intérêt de la méthode d'Euler est qu'elle peut-être utilisée pour la détermination d'une solution approchée d'une équation différentielle dont la solution analytique exacte est difficile ou impossible à obtenir. On montre ci-dessous la solution approchée par la méthode d'Euler pour une excitation sinusoïdale : on observe bien un régime transitoire suivi d'un régime permanent sinusoïdal forcé.

Manipulation 4
Modifier le code précédent de façon à reproduire la courbe ci-dessous. Vous ferez apparaitre les solutions approchées calculées avec des pas de $0,01\tau$, $0,1\tau$,$0,5\tau$ et τ (vous pourrez en revanche omettre la solution exacte).



IV - Utilisation de la fonction odeint

Pour éviter d'avoir à recoder à chaque fois la méthode d'Euler explicite, il est possible d'utiliser la fonction `odeint` du module `scipy.integrate`, qui permet de résoudre une équation différentielle du premier ordre de la forme $y' = f(y, t)$. On notera cependant que cette fonction utilise une méthode beaucoup plus efficace que la méthode d'Euler, ce qui permet, pour une valeur du pas δt donnée, d'avoir une solution appochée beaucoup plus proche de la solution exacte. La syntaxe de la fonction odeint est légèrement différente de celle de la fonction euler que vous avez écrite précédemment.
Parcourir rapidement la documentation pour comprendre son fonctionnement :

- http://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.odeint.html
- ou bien en console help(odeint)

Manipulation 5
Faire un nouveau graphe qui représente la réponse du circuit à un échelon de tension, où vous superposez, pour un pas de temps donné, les solutions exacte, obtenue avec la fonction `euler` et obtenue avec la fonction `odeint`.

V - Modélisation d'un onduleur

Afin d'alimenter une installation électrique, la puissance continue délivrée par un panneau photovoltaïque doit être convertie en puissance alternative. Cette conversion est réalisée au moyen d'un onduleur, comme schématisé sur la figure 1.1. On cherche à modéliser le signal de sortie s fourni par cet onduleur à l'installation, assimilée à une charge résistive R, ainsi qu'à observer l'influence de la séquence de commande des interrupteurs sur la forme du signal s.

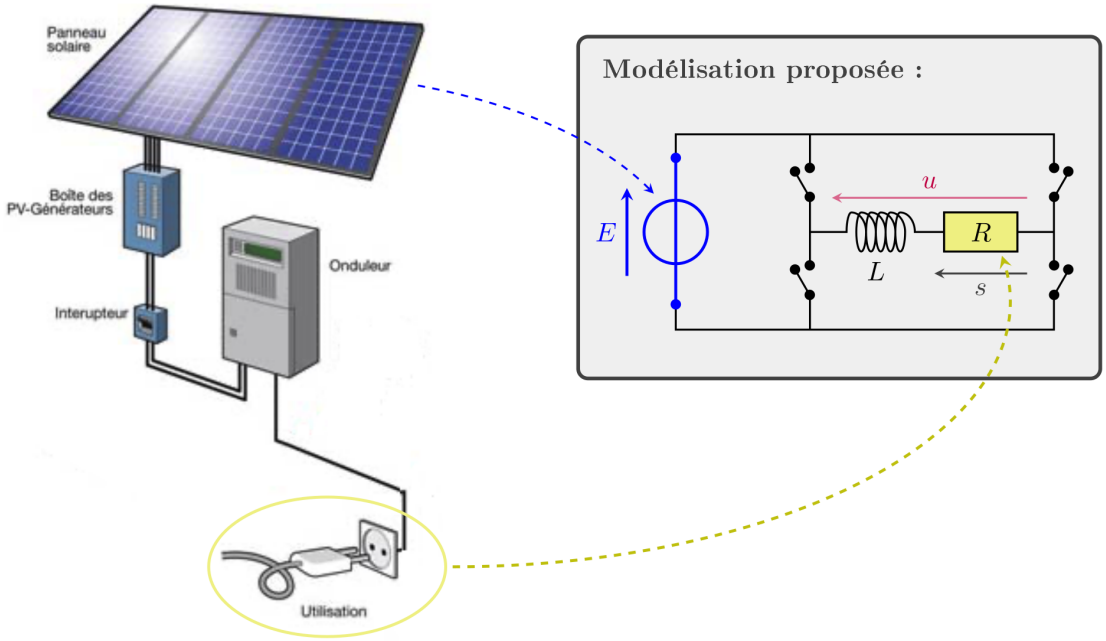


FIGURE 1.1 – Raccordement d'un panneau photovoltaïque à une installation domestique via un onduleur : schéma de principe et modélisation.

1. Montrer que les tensions $u(t)$ et $s(t)$ sont liées par l'équation

$$\forall t,\ \frac{\mathrm{d}s}{\mathrm{d}t} + \frac{s(t)}{\tau} = \frac{u(t)}{\tau}$$

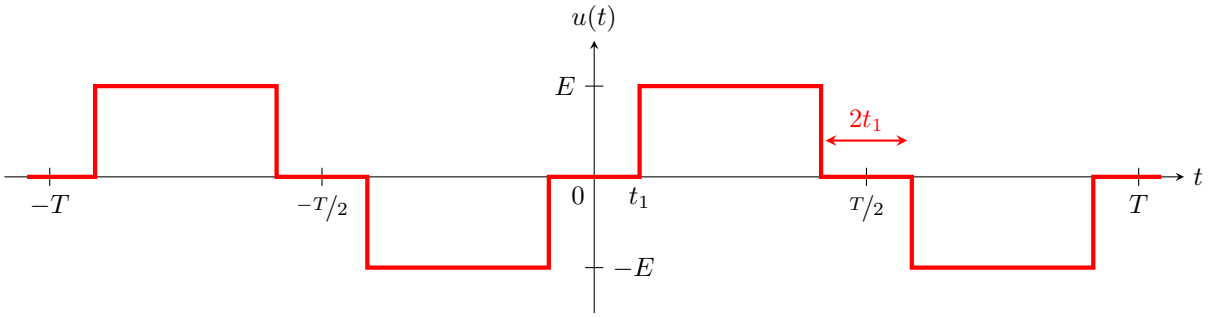
   Préciser l'expression du paramètre τ et en donner une interprétation physique.

2. On suppose dans un premier temps que la séquence de commande des interrupteurs donne un signal $u(t)$ en forme de créneau, symétrique, d'amplitude E et de période T. On souhaite déterminer le signal $s(t)$ correspondant en résolvant numériquement l'équation différentielle établie dans la question précédente.

   (a) En admettant que $f = 1/T = 50\,\mathrm{Hz}$ et que l'on choisit L et R de sorte à avoir $\tau = T/2\pi$, proposer une valeur du pas de discrétisation adaptée a priori à la résolution numérique du problème.

   (b) Compléter le script disponible sur le site de la PTSI (texttCN-TP2-Euler-onduleur-start) de façon à implémenter la méthode d'Euler explicite de résolution d'une équation différentielle et résoudre numériquement l'équation différentielle donnant l'évolution de la tension $s(t)$ sur l'intervalle $[0,\ 4T]$, avec la condition initiale $s(0) = 0$. Tracer, sur un même graphe, les courbes représentatives des tensions u et s. Commenter l'allure de la tension s ainsi déterminée.

3. On modifie la séquence de commande des interrupteurs de sorte à ce que le signal créneau $u(t)$ présente des paliers nuls de durée $2t_1$, avec $t_1 = T/12$, comme sur le chronogramme ci-dessous.



(a) Dans votre script python, adapter la définition de la fonction $u(t)$ de façon à prendre en compte cette modification, puis reprendre la résolution numérique de l'équation différentielle vérifiée par la tension $s(t)$.

(b) On propose de modéliser le signal $s(t)$ ainsi obtenu par le signal sinusoïdal

$$s_{\text{mod}} = \frac{\sqrt{6}E}{\pi} \sin\left(2\pi f\left(t - \frac{T}{8}\right)\right)$$

Ajouter au script python le tracé de cette fonction. Visualiser graphiquement la pertinence d'une telle modélisation.

On définit le rendement énergétique de l'onduleur par

$$\eta = \frac{\text{puissance utile}}{\text{puissance fournie}} = \frac{\langle s(t) \times i(t)\rangle}{\langle u(t) \times i(t)\rangle} = \frac{\langle s^2(t)\rangle}{\langle u(t) \times s(t)\rangle} \approx \frac{\langle s_{\text{mod}}^2(t)\rangle}{\langle u(t) \times s_{\text{mod}}(t)\rangle}$$

où $\langle\ \rangle$ est le symbole pour la valeur moyenne temporelle.

Pour le calcul des valeurs moyennes, on peut calculer un grand nombre de valeurs de la tension s_{mod} sur une période, puis utiliser la fonction `mean()` de la bibliothèque `numpy` (ce qui revient à approcher numériquement les intégrales intervenant dans la définition des valeurs moyennes par la méthode des rectangles à gauche).

(c) Estimer le rendement énergétique de l'onduleur.