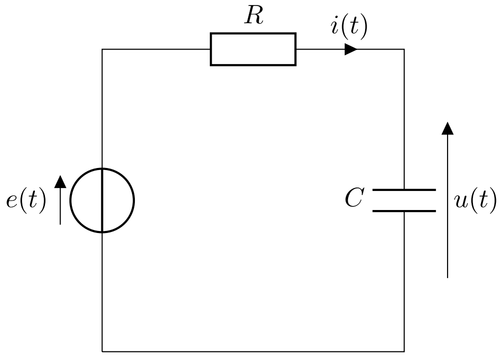
Capacité numérique : la méthode d'Euler

On présente ici une **méthode numérique** – la méthode d'Euler – permettant de simuler la réponse d'un **système linéaire du premier ordre** à une **excitation quelconque.**



On considère par exemple le circuit RC ci-contre. La tension $u(t)$ vérifie l'équation différentielle suivante :

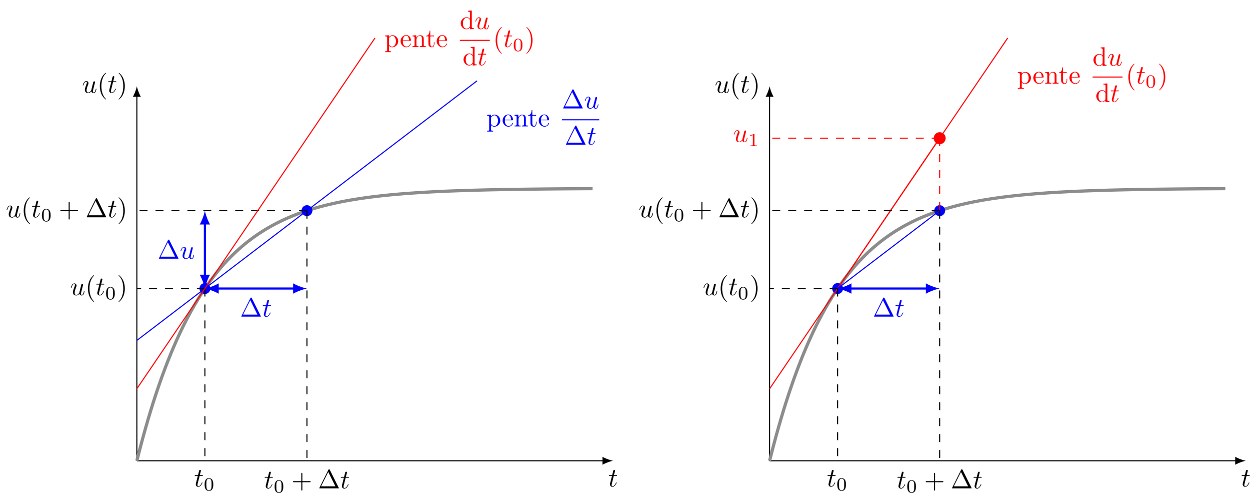
$$\frac{\mathrm{d}u}{\mathrm{d}t}(t) + \frac{1}{\tau}u(t) = \frac{1}{\tau}e(t) \;, \tag{1}$$

avec $\tau = RC$ la constante de temps du circuit. Dans les **cas simples,** par exemple si $e(t)$ est constante par morceaux (cf. Chapitre 6) ou une fonction sinusoïdale (cf. Chapitre 8), on sait résoudre analytiquement cette équation.

Néanmoins la résolution analytique peut parfois s'avérer **fastidieuse** et on se tourne alors naturellement vers une **résolution numérique.**

I Principe de la méthode

1) Approximation de la dérivée



☞ **Interprétation graphique de la dérivée :** la pente de la **tangente** à la courbe représentative de la fonction $u(t)$ **à l'instant** t_0 est égale à $\frac{\mathrm{d}u}{\mathrm{d}t}(t_0)$ (dérivée de la fonction calculée à l'instant t_0).

☞ **Approximation de la dérivée :**

$$\frac{\mathrm{d}u}{\mathrm{d}t}(t_0) \simeq \frac{\Delta u}{\Delta t} = \frac{u(t_0+\Delta t) - u(t_0)}{\Delta t} \quad \Rightarrow \quad \boxed{u(t_0+\Delta t) \simeq u(t_0) + \Delta t \times \frac{\mathrm{d}u}{\mathrm{d}t}(t_0) \overset{\text{def.}}{=} u_1} \,. \tag{2}$$

2) Exemple

On essaye de résoudre de **manière approchée** le problème de Cauchy suivant :

$$\begin{cases} \dfrac{\mathrm{d}u}{\mathrm{d}t}(t) + \dfrac{1}{\tau}u(t) = \dfrac{1}{\tau}E \;, \quad \text{pour} \quad t \geq t_0 \;, \\ u(t_0) = u_0 \;. \end{cases} \tag{3}$$

Concrètement, on cherche à calculer les **valeurs approchées** de la fonction u solution de l'équation aux instants $t_1, t_2, \ldots t_{n-1}$ avec $t_{k+1} = t_k + \Delta t$. On prendra $E > 0$, $t_0 = 0$, $u_0 = 0$ et $\Delta t = 3\tau/4$ pour l'exemple (FIGURE 1).

Évidemment on sait résoudre cette équation de façon exacte :

$$u(t) = E(1 - e^{-t/\tau}) \ , \quad t \geq 0 \ . \tag{4}$$

On se servira de la solution exacte pour tester la méthode !

1ère étape : on connait **exactement** la valeur $u(t_0)$ de la fonction u en $t = t_0$, c'est la condition initiale. La condition initiale permet aussi de calculer la **valeur exacte** de $\frac{\mathrm{d}u}{\mathrm{d}t}(t_0)$:

$$\frac{\mathrm{d}u}{\mathrm{d}t}(t_0) = \big(E - u(t_0)\big)/\tau = E/\tau \ . \tag{5}$$

On se sert de l'approximation de la dérivée dans l'équation (2) pour calculer la valeur **approchée** u_1 de $u(t_1)$:

$$u(t_1) \simeq u(t_0) + \Delta t \times \frac{\mathrm{d}u}{\mathrm{d}t}(t_0) \ \stackrel{\text{def.}}{=} \ u_1 \ . \tag{6}$$

2ème étape : on se sert à nouveau de l'équation (2) pour calculer la valeur **approchée** u_2 de $u(t_2)$:

$$u(t_2) \simeq u(t_1) + \Delta t \times \frac{\mathrm{d}u}{\mathrm{d}t}(t_1) \ . \tag{7}$$

Cette fois on ne connaît ni la valeur exacte de $u(t_1)$ ni celle de $\frac{\mathrm{d}u}{\mathrm{d}t}(t_1) = \big(E - u(t_1)\big)/\tau$. On utilise alors la valeur approchée u_1 de $u(t_1)$ et l'équation différentielle (3) pour calculer la **valeur approchée** de $\frac{\mathrm{d}u}{\mathrm{d}t}(t_1)$:

$$\frac{\mathrm{d}u}{\mathrm{d}t}(t_1) \simeq (E - u_1)/\tau \ , \tag{8}$$

puis la **valeur approchée** u_2 de $u(t_2)$:

$$u(t_2) \simeq u_1 + \Delta t \times (E - u_1)/\tau \ \stackrel{\text{def.}}{=} \ u_2 \ . \tag{9}$$

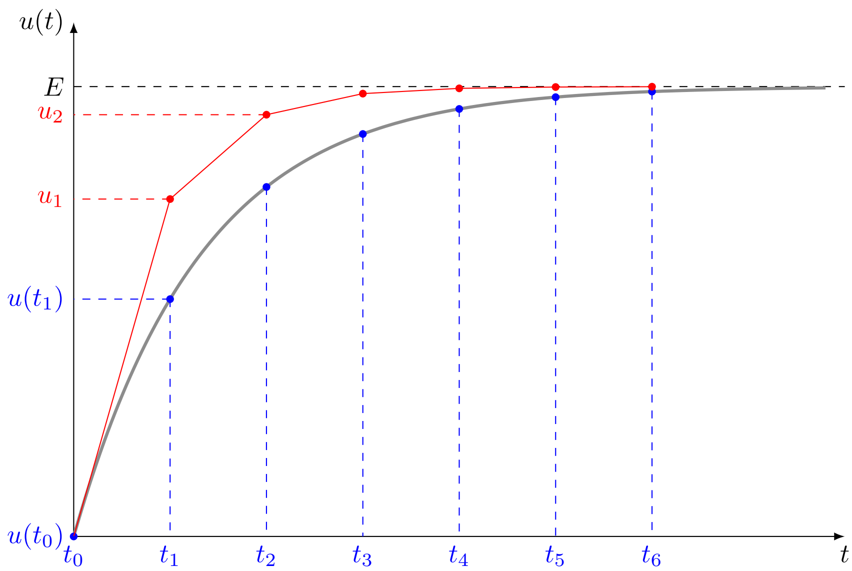


Figure 1. Résolution approchée de l'équation (3) et comparaison à la solution exacte. On a pris $\Delta t = 3\tau/4$.

3ème étape, 4ème étape, etc. : on applique la même formule, **pas à pas.**

$$\begin{aligned}
u(t_3) &\simeq u_2 + \Delta t \times (E - u_2)/\tau \ \stackrel{\text{def.}}{=} \ u_3 \ , \\
u(t_4) &\simeq u_3 + \Delta t \times (E - u_3)/\tau \ \stackrel{\text{def.}}{=} \ u_4 \ , \\
&\vdots \\
u(t_{n-1}) &\simeq u_{n-2} + \Delta t \times (E - u_{n-2})/\tau \ \stackrel{\text{def.}}{=} \ u_{n-1} \ ,
\end{aligned}$$

En **réduisant le pas** Δt on **améliore la précision** de la méthode (FIGURE 2)

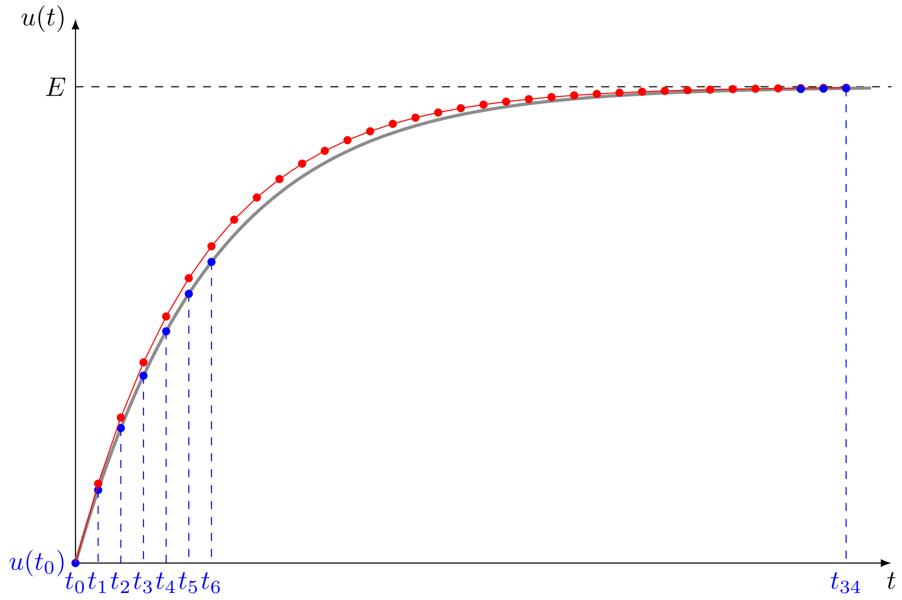


FIGURE 2. Résolution approchée de l'équation (3) et comparaison à la solution exacte. On a pris $\Delta t = \tau/6$.

Au contraire, en **augmentant le pas** Δt on peut obtenir des résultats aberrants **du point de vue physique,** comme un dépassement de la valeur limite et des oscillations (FIGURE 3), caractéristiques d'un système du 2ème ordre !

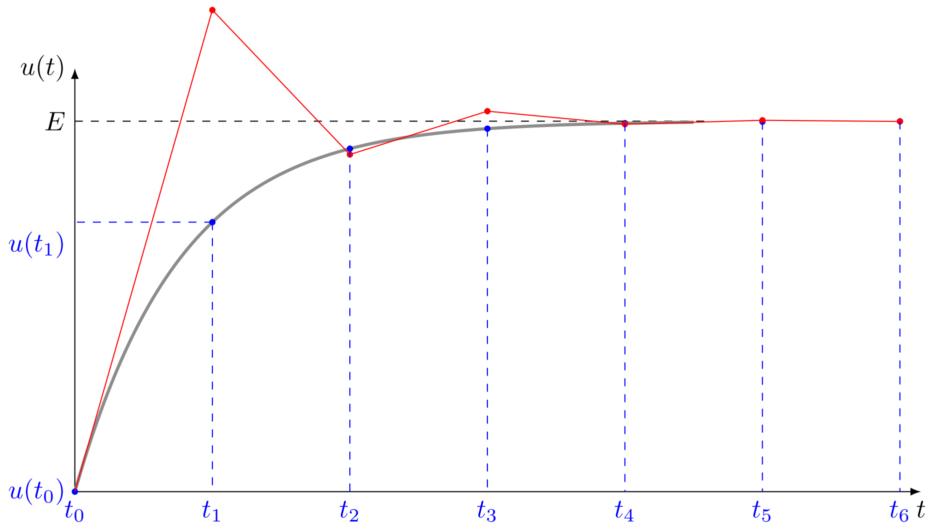


FIGURE 3. Résolution approchée de l'équation (3) et comparaison à la solution exacte. On a pris $\Delta t = 1,3\tau$.

II Mise en œuvre concrète

1) L'algorithme

Méthode d'Euler.

☞ On s'intéresse à la résolution approchée du problème de Cauchy suivant :

$$\begin{cases} \dfrac{\mathrm{d}u}{\mathrm{d}t}(t) = f(u(t), t) \ , \quad \text{pour} \quad t \geq t_0 \ , \\ u(t_0) = u_0 \ . \end{cases} \tag{10}$$

☞ Dans le cas d'un **système linéaire du premier ordre** :

$$\boxed{f(u(t),t) = e(t)/\tau - u(t)/\tau}\,, \tag{11}$$

avec $\tau > 0$ la **constante de temps** du système et $e(t)$ une fonction quelconque du temps, représentant **l'excitation** du système.

☞ On cherche à calculer les **valeurs approchées** de la fonction u solution de l'équation différentielle aux instants $t_1, t_2, \dots t_{n-1}$ avec $t_{k+1} = t_k + \Delta t$.

☞ Ces valeurs sont données par la formule de récurrence :

$$\boxed{u(t_{k+1}) \simeq u_{k+1} \overset{\text{def.}}{=} u_k + \Delta t \times f(u_k, t_k)}\,. \tag{12}$$

2) Le code Python

Fonction python :

On choisit d'écrire une **fonction Python** `euler(f,u0,t)` permettant de résoudre le problème de Cauchy (10) à l'aide de la méthode d'Euler.

☞ Cette fonction prend **3 paramètres :**

— une fonction `f` de 2 variables, qui donne la dérivée de la fonction u ;

— un nombre `u0`, qui représente la condition initiale $u(t_0) = u_0$;

— un tableau numpy `t`, qui représente l'ensemble des instants $\{t_0, t_1, \dots, t_{n-1}\}$ auxquels on cherche à évaluer la fonction u .

☞ Le **code** de la fonction :

```python
1  def euler(f,u0,t):
2      """
3      Fonction implémentant la méthode d'Euler pour l'équation différentielle du/dt = f(u(t),t)
4  
5      Entrée
6      -------------------------------
7      f : fonction donnant du/dt (fonction de 2 variables)
8      u0 : condition initiale u(t_0) = u_0 (float)
9      t : tableau numpy représentant les instants (t_0,t_1,...,t_n-1) auxquels évaluer u(t)
10 
11     Sortie
12     -------------------------------
13     tableau numpy représentant les u(t_k)
14     """
15     n = len(t)
16     dt = t[1]-t[0]
17     sol = np.zeros(n)
18     sol[0] = u0
19     for k in range(n-1):
20         sol[k+1] = sol[k] + dt*f(sol[k],t[k])
21     return sol
```

La ligne 1 définit la fonction `euler`. La ligne 15 extrait le nombre n d'instants contenus dans la liste `t` et le stocke dans la variable `n`, la ligne 16 extrait le pas de temps Δt et le stocke dans la variable `dt`. La ligne 17 crée un tableau numpy `sol` de longueur `n`, rempli de zéros, dans lequel on va stocker les valeurs u_k. La ligne 18 stocke la condition initiale $u(t_0) = u_0$ dans le premier élément du tableau `sol`. Les lignes 19 et 20 implémentent la formule de récurrence (12) en remplissant le tableau `sol`. La ligne 21 renvoi le tableau `sol`.

3) Comment choisir le pas de temps et le nombre de points à calculer ?

D'après l'équation (2), l'approximation de la dérivée s'améliore lorsque $\Delta t \to 0$. Il faut donc choisir le pas de temps Δt suffisamment petit. Petit par rapport à quoi ? Pour répondre à cette question il faut identifier la ou les **échelles de temps naturelles** du système.

Choix du pas de temps :

☞ Pour un **système linéaire du premier ordre**, les échelles de temps naturelles sont :

— la constante de temps τ du système ;

— la durée typique T de variation de l'excitation $e(t)$.

Le pas de temps Δt doit être choisi tel que $\Delta t \ll \tau$ **et** $\Delta t \ll T$.

Pour un signal périodique $e(t)$, la durée typique T est simplement la période.

Choix de l'instant final :

On applique la méthode d'Euler entre l'instant initial t_0 et et l'instant final $t_f = t_{n-1} = t_0 + (n-1) \times \Delta t$.

☞ Il faut choisir l'instant final tel que $t_f - t_0 \gg \tau$ **et** $t_f - t_0 \gg T$, afin d'observer toutes les variations importantes de la fonction $u(t)$.

En pratique on définit un tableau des instants $\{t_0, t_1, \ldots, t_{n-1}\}$ de la forme `t = np.linspace(t0,tf,n)`, contenant `n` éléments régulièrement espacés de $\Delta t = (t_f - t_0)/(n-1)$. Le premier élément `t[0]` vaut `t0` et le dernier élément `t[n-1]` vaut `tf`. Il faut donc choisir :

— `tf` assez grand pour que $t_f - t_0 \gg \tau$, T ;

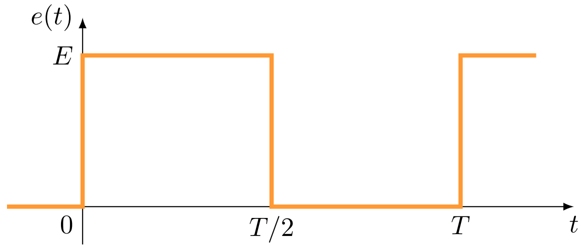
— `n` assez grand pour que $\Delta t \ll \tau$, T .

Pour fixer la valeur de n il faut procéder par essai/erreur : on choisit une première valeur de n, puis on multiplie par 2 successivement jusqu'à ce que la fonction $u(t)$ ne varie plus de façon appréciable.

4) Application

On soumet le circuit RC de l'équation (1) à une excitation créneau $[0, E]$ de période T. La condition initiale est $u(t = 0) = 0$ et on résout l'équation différentielle à l'aide de la méthode d'Euler.

Voici un exemple de code qui implémente la méthode et trace le graphe des fonctions.




```python
##Bibliothèques utiles
import numpy as np
import matplotlib.pyplot as plt

## Fonction euler
def euler(f,u0,t):
    """
    Fonction implémentant la méthode d'Euler pour l'équation différentielle du/dt = f(u(t),t)

    Entrée
    -------------------------------
    f : fonction donnant du/dt (fonction de 2 variables)
    u0 : condition initiale u(t_0) = u_0 (float)
    t : tableau numpy représentant les instants (t_0,t_1,...,t_n-1) auxquels évaluer la
    fonction u

    Sortie
    -------------------------------
    tableau numpy représentant les u(t_k)
    """
    n = len(t)
    dt = t[1]-t[0]
    sol = np.zeros(n)
    sol[0] = u0
    for k in range(n-1):
        sol[k+1] = sol[k] + dt*f(sol[k],t[k])
    return sol
```

```python
### Définition des paramètres du circuit
tau = 1e-4 #s   (constante de temps du circuit)
T = 2e-3 #s (--> Fig.4) (période de l'excitation créneau)
#T = 1e-5 #s (--> Fig.5) (période de l'excitation créneau)
E = 10. #V (amplitude de l'excitation créneau)

### Définition de l'excitation créneau
def creneau(t):
    """
    Fonction créneau de période T variant entre 0 et E

    Entrée
    -------------------------------
    t : valeur de l'instant auquel est évaluée la fonction (float)

    Sortie
    -------------------------------
    valeur de la fonction (float)
    """
    if t%T < T/2 :  #t%T est le reste de la division euclidienne de t par T
        return E
    else :
        return 0.

### Définition de la fonction donnant la dérivée de la tension u
def f(u,t):
    """
    Fonction donnant du/dt (fonction de 2 variables)

    Entrée
    -------------------------------
    u : valeur de u(t) (float)
    t : valeur de l'instant t (float)

    Sortie
    -------------------------------
    valeur de la dérivée du/dt (float)
    """
    dudt = (creneau(t) - u)/tau
    return dudt

### Définition du tableau t des instants auxquels évaluer la fonction
t0 = 0.    # instant initial t_0
tf = 3*T  # instant final t_{n-1}  ---> Figure 4
#tf = 60*T # instant final t_{n-1}  ---> Figure 5
n = 4000 # nombre d'instants
t = np.linspace(t0,tf,n) # tableau contenant les instants régulièrement espacés (t_0,t_1,...,t_n-1) auxquels évaluer la fonction u

### Définition de la condition initiale
u0 = 0 # V (condition intiale u(t0) = u0)

### Résolution numérique de l'équation différentielle
solution = euler(f,u0,t) # tableau contenant les n valeurs u_k

### Graphe
tgraphe = np.linspace(t0,tf,4000)
e = np.array([creneau(temps) for temps in tgraphe]) #tableau contenant les valeurs e(t) de l'excitation

plt.figure()
plt.plot(tgraphe,e,color='orange', label = '$e(t)$',linewidth=2) #graphe de e(t)
plt.plot(t,solution,'-or',label = 'u_c(t)',markersize = 2, linewidth=1) #graphe de u(t)
plt.xlabel('$t$ (s)')
plt.ylabel('tensions (V)')
plt.legend(loc=1)
plt.tight_layout()
plt.grid(linewidth=0.2)
plt.show()
```

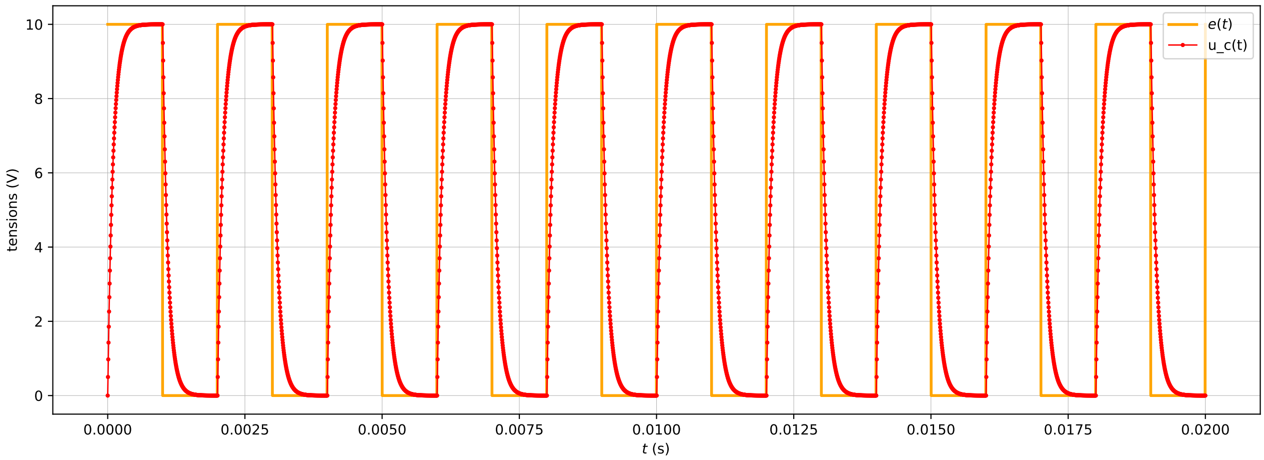


Figure 4. Réponse du circuit RC série à une excitation créneau de période $T = 2\,\text{ms}$, avec $\tau = 0{,}1\,\text{ms}$ soit $\tau = T/20$.

Pour l'exemple de la Figure 4 on a pris $T = 2\,\text{ms}$, avec $\tau = 0{,}1\,\text{ms}$ soit $T > \tau$. Il faut donc choisir l'instant final tel que $t_f \gg T$ et le nombre n de points tel que $\Delta t = (t_f - t_0)/(n-1) \ll \tau$. Avec $t_f = 10T$ ($t_f \gg T$) et $n = 4000$ on trouve $\Delta t \simeq 0{,}0025\,\text{ms} \ll \tau$.

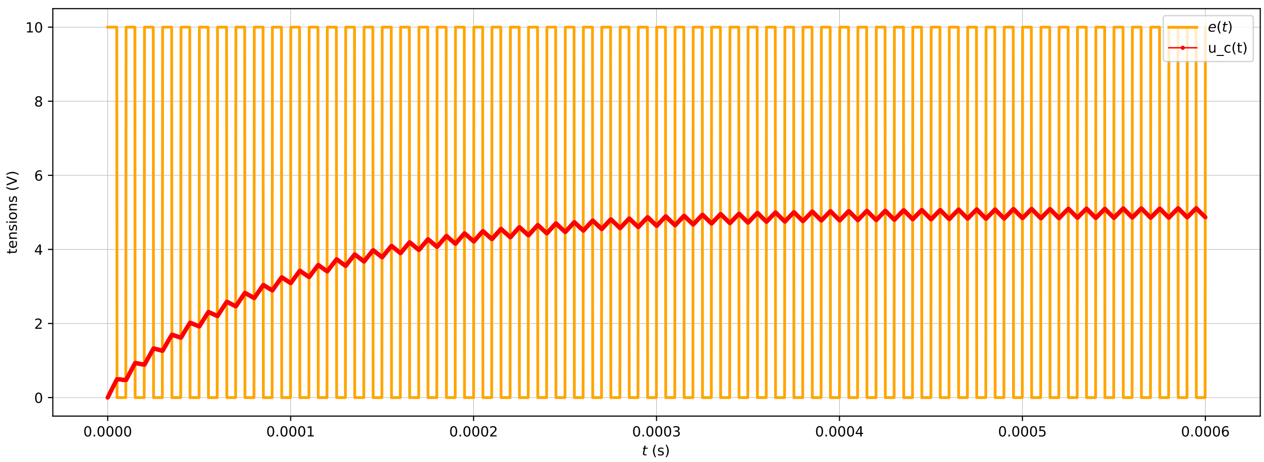


Figure 5. Réponse du circuit RC série à une excitation créneau de période $T = 0{,}01\,\text{ms}$, avec $\tau = 0{,}1\,\text{ms}$ soit $\tau = 10 \times T$.

Pour l'exemple de la Figure 5 on a pris $T = 0{,}01\,\text{ms}$, avec $\tau = 0{,}1\,\text{ms}$ soit $\tau > T$. Il faut donc choisir l'instant final tel que $t_f \gg \tau$ et le nombre n de points tel que $\Delta t = (t_f - t_0)/(n-1) \ll T$. Avec $t_f = 60T$, on trouve $t_f = 0{,}6\,\text{ms}$ grand devant τ (mais on pourrait augmenter t_f par sécurité). De plus avec $n = 4000$ on trouve $\Delta t \simeq 0{,}000\,15\,\text{ms} \ll T$.