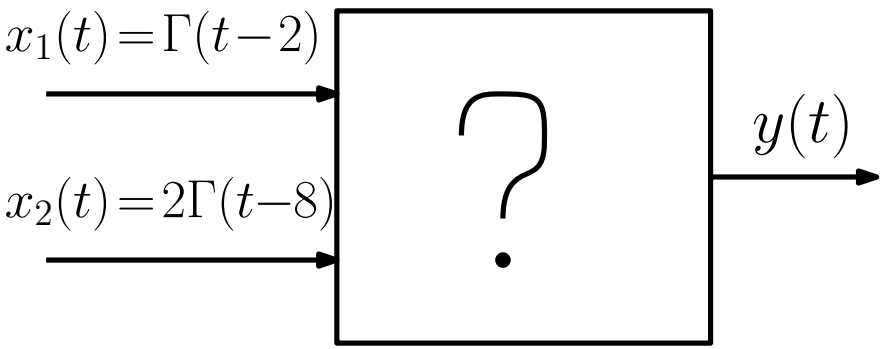
Outils Mathématiques pour l'Ingénieur
...
TD3 de Traitement du Signal
...
(a) Identification
(b) Simulation de systèmes linéaires et non linéaires

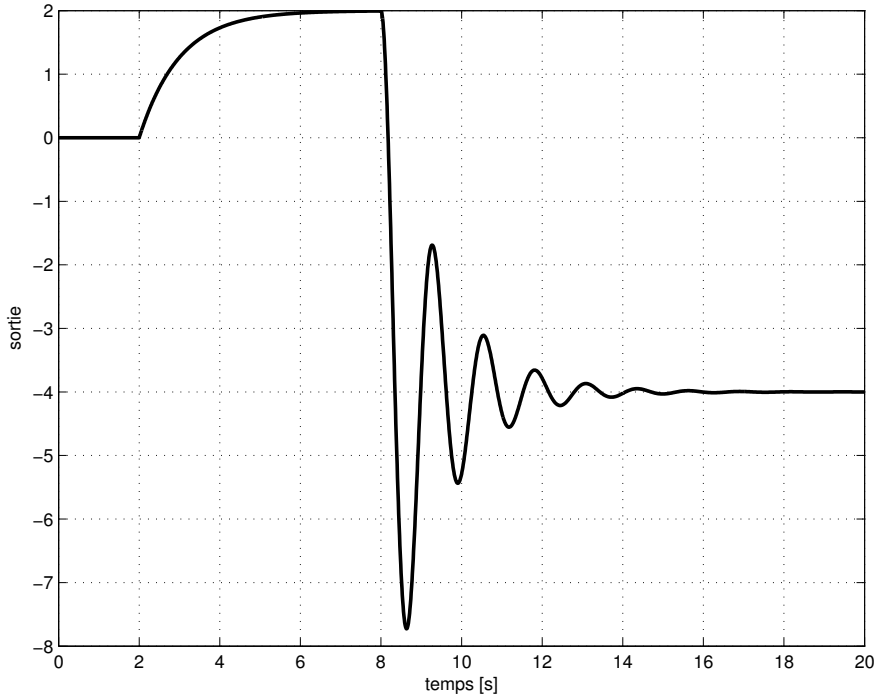
Benoît Marx

Centre de Recherche en Automatique de Nancy (CRAN)
Ecole Nationale Supérieure de Géologie (ENSG)

(a) Identification

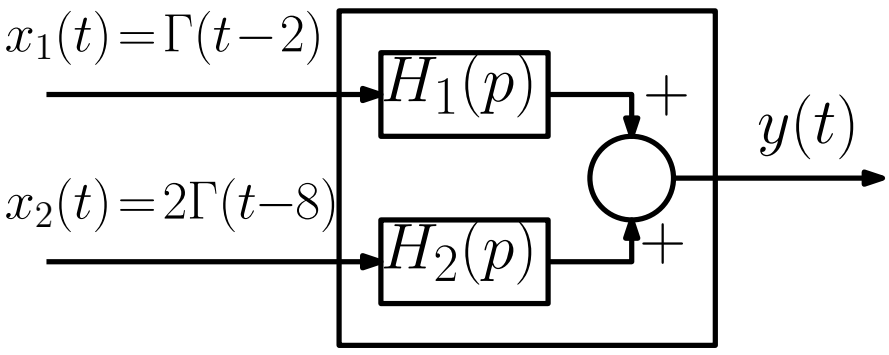
- **Objectif :** mesures entrées/sortie $\Rightarrow$ **modèle** du système
- **Données :** fichier `dataTD31.mat` sur Arche

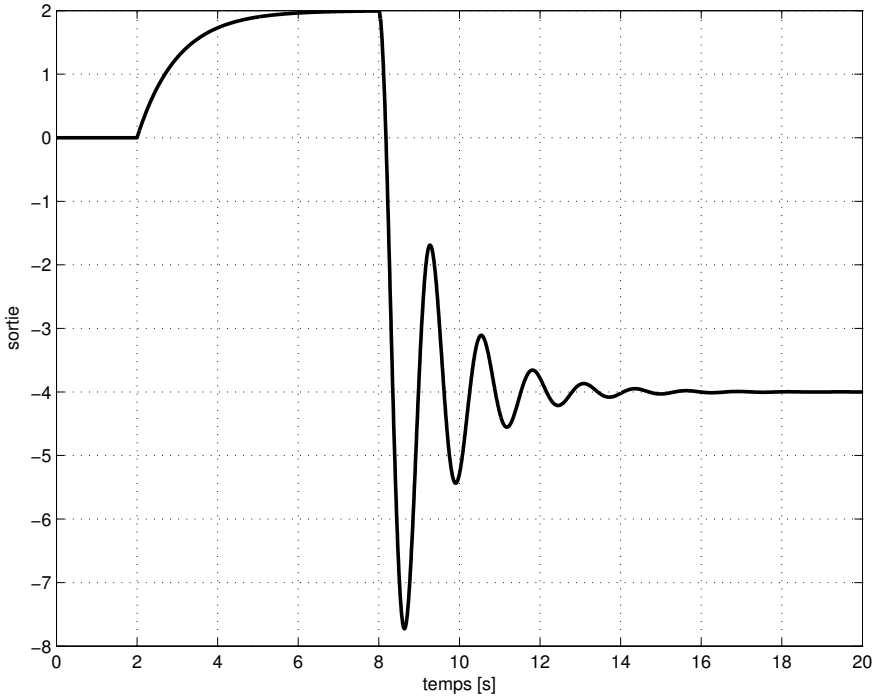




(a) Identification

- **Objectif :** mesures entrées/sortie $\Rightarrow$ **modèle** du système
- **Données :** fichier `dataTD31.mat` sur Arche



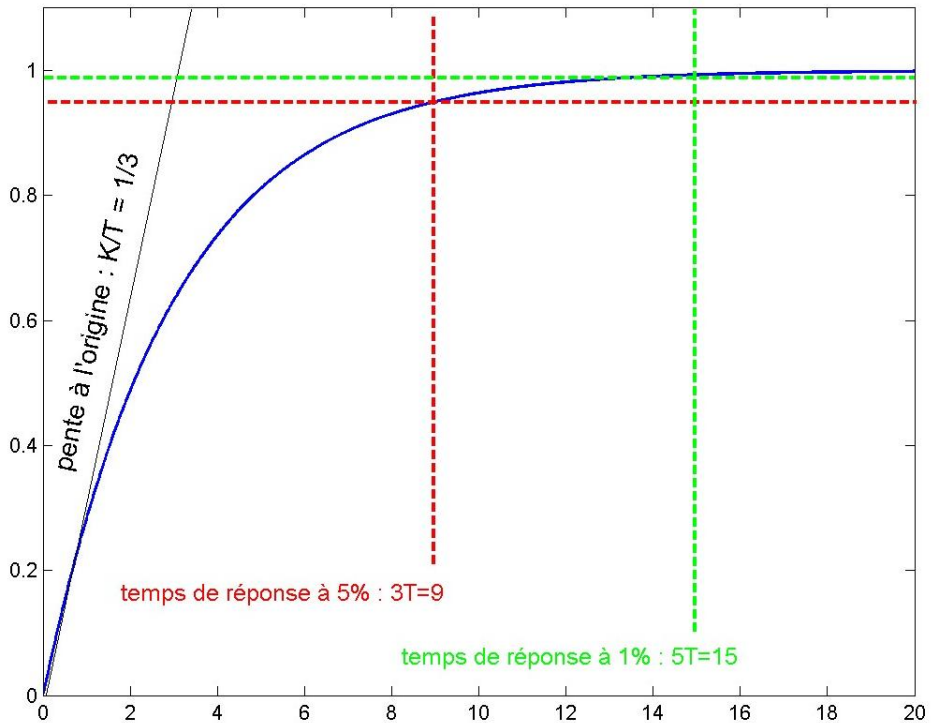


(a) Identification

Rappels : réponse indicielle ($x(t) = \Gamma(t)$) d'un système du 1^{er} ordre

$$y(t) = K(1 - e^{-t/T})$$

- valeur finale : $\lim_{t\to\infty} y(t) = K$
- valeur initiale de la pente : $\dot{y}(0) = K/T$
- temps de réponse à 5% : $t_{5\%} \approx 3T$
- temps de réponse à 2% : $t_{2\%} \approx 4T$
- temps de réponse à 1% : $t_{1\%} \approx 4.6T$



Temps de réponse à $x\%$: instant $t_{x\%}$ tel que, pour tout $t \geq t_{x\%}$:

$$y\left(\infty\right)\left(1 - \tfrac{x}{100}\right) \leq y(t) \leq \left(\infty\right)\left(1 + \tfrac{x}{100}\right)$$

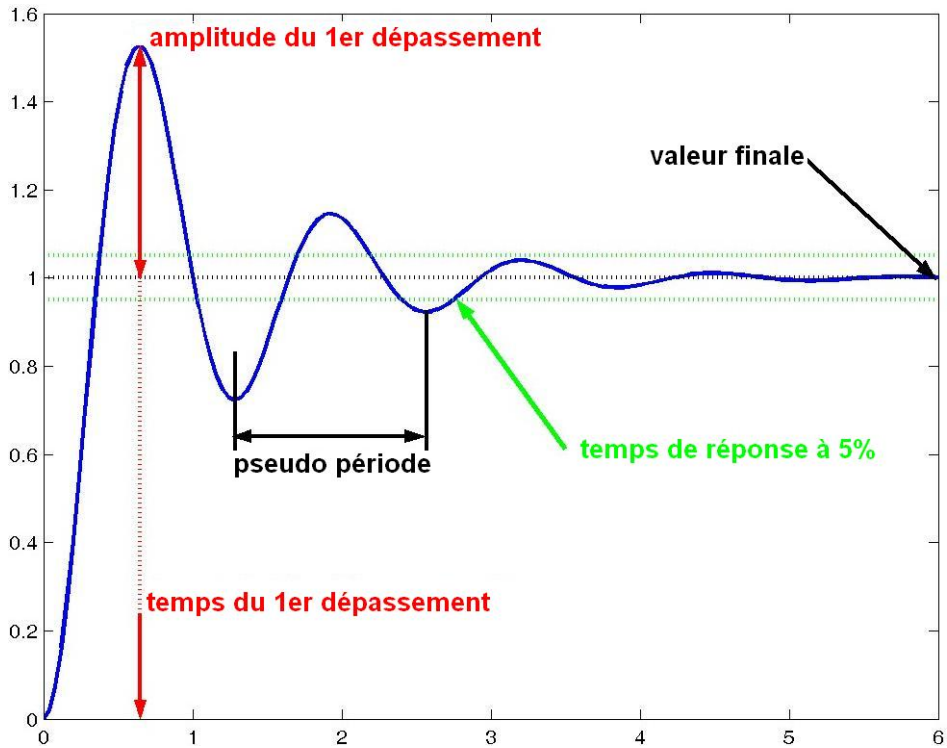
(a) Identification

Détermination de $H_1(p)$ d'ordre 1

Rappels : réponse indicielle ($x(t) = \Gamma(t)$) d'un système du $2^{ème}$ ordre, avec $z < 1$

$$y(t) = K\left(1 - \frac{e^{-z\omega_0 t}}{\sqrt{1-z^2}}\sin(\omega_0\sqrt{1-z^2}t + \phi)\right), \ pour \ \phi = Atan\left(\frac{\sqrt{1-z^2}}{z}\right)$$

- valeur finale :
$$\lim_{t\to\infty} y(t) = K$$
- pseudo-période :
$$T_n = \frac{2\pi}{\omega_0\sqrt{1-z^2}}$$
- temps de réponse à 5% :
$$t_{5\%} \approx \frac{3}{z\omega_0}$$
- amplitude du 1^{er} dépassement :
$$X_p = Ke^{-\frac{\pi z}{\sqrt{1-z^2}}}$$
- instant du 1^{er} dépassement :
$$t_p = \frac{\pi}{\omega_0\sqrt{1-z^2}}$$



Détermination de $H_2(p)$ d'ordre 2

Validation en simulation avec Matlab

Validation en simulation avec Matlab

- Définir les fonctions de transfert (fonction `tf(...,...)`) :
- Définir les entrée $x_1(t) = \Gamma(t-2)$ et $x_2(t) = 2\Gamma(t-8)$:
- Simuler les réponses de $H_1(p)$ et $H_2(p)$ (fonction `lsim(H1,x1,t)`):
- Tracer les sorties mesurée et simulée :

Validation en simulation avec Matlab

- Définir les fonctions de transfert (fonction `tf(...,...)`) :

```
K1=... ;T= ... ;
K2=... ;w0= ... ;z= ... ;
H1=tf(K1,[T 1]);
H2=tf(K2,[1/w0^2 2*z/w0 1]);
```

- Définir les entrée $x_1(t) = \Gamma(t-2)$ et $x_2(t) = 2\Gamma(t-8)$:

- Simuler les réponses de $H_1(p)$ et $H_2(p)$ (fonction `lsim(H1,x1,t)`):

- Tracer les sorties mesurée et simulée :

Validation en simulation avec Matlab

- Définir les fonctions de transfert (fonction `tf(...,...)`) :

```
K1=... ;T= ... ;
K2=... ;w0= ... ;z= ... ;
H1=tf(K1,[T 1]);
H2=tf(K2,[1/w0^2 2*z/w0 1]);
```

- Définir les entrée $x_1(t) = \Gamma(t-2)$ et $x_2(t) = 2\Gamma(t-8)$:

```
x1=[zeros(1,200) ones(1,1801)];
x2=[zeros(1,800) 2*ones(1,1201)];
```

- Simuler les réponses de $H_1(p)$ et $H_2(p)$ (fonction `lsim(H1,x1,t)`):

- Tracer les sorties mesurée et simulée :

Validation en simulation avec Matlab

- Définir les fonctions de transfert (fonction `tf(...,...)`) :

```
K1=... ;T= ... ;
K2=... ;w0= ... ;z= ... ;
H1=tf(K1,[T 1]);
H2=tf(K2,[1/w0^2 2*z/w0 1]);
```

- Définir les entrée $x_1(t) = \Gamma(t-2)$ et $x_2(t) = 2\Gamma(t-8)$:

```
x1=[zeros(1,200) ones(1,1801)];
x2=[zeros(1,800) 2*ones(1,1201)];
```

- Simuler les réponses de $H_1(p)$ et $H_2(p)$ (fonction `lsim(H1,x1,t)`):

```
[y1]=lsim(H1,x1,t);
[y2]=lsim(H2,x2,t);
```

- Tracer les sorties mesurée et simulée :

Validation en simulation avec Matlab

- Définir les fonctions de transfert (fonction `tf(...,...)`) :

```
K1=... ;T= ... ;
K2=... ;w0= ... ;z= ... ;
H1=tf(K1,[T 1]);
H2=tf(K2,[1/w0^2 2*z/w0 1]);
```

- Définir les entrée $x_1(t) = \Gamma(t-2)$ et $x_2(t) = 2\Gamma(t-8)$:

```
x1=[zeros(1,200) ones(1,1801)];
x2=[zeros(1,800) 2*ones(1,1201)];
```

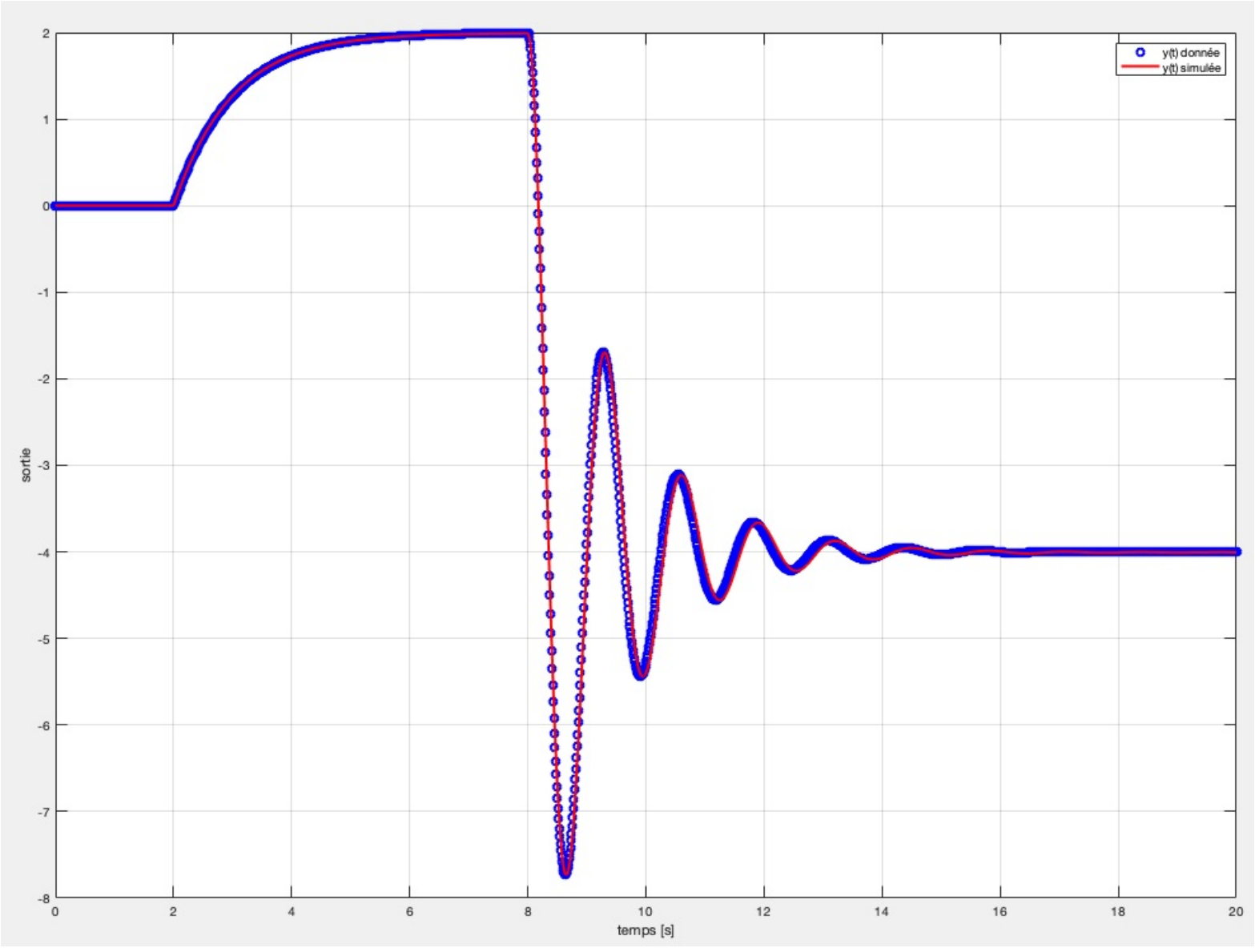
- Simuler les réponses de $H_1(p)$ et $H_2(p)$ (fonction `lsim(H1,x1,t)`):

```
[y1]=lsim(H1,x1,t);
[y2]=lsim(H2,x2,t);
```

- Tracer les sorties mesurée et simulée :

```
hold on
plot(t,y,'bo')
plot(t,y1+y2,'r')
```

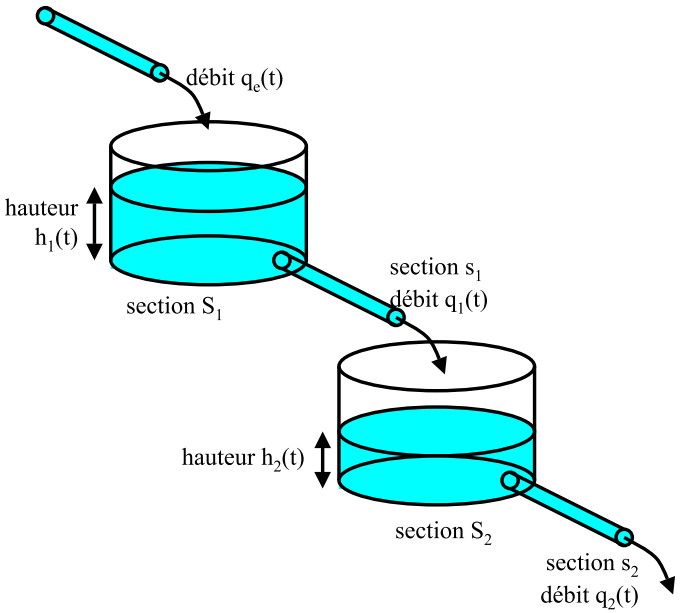
(a) Identification, validation des résultats



(b) Linéarisation et simulation du système linéaire

- **Données :** fichier `dataTD32.mat` sur Arche

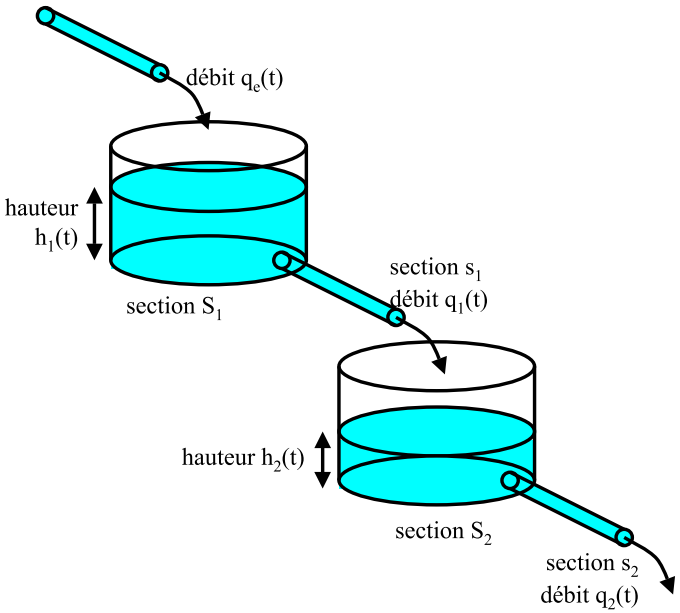
Système à modéliser



(b) Linéarisation et simulation du système linéaire

- **Données :** fichier `dataTD32.mat` sur Arche

Système à modéliser



Modélisation

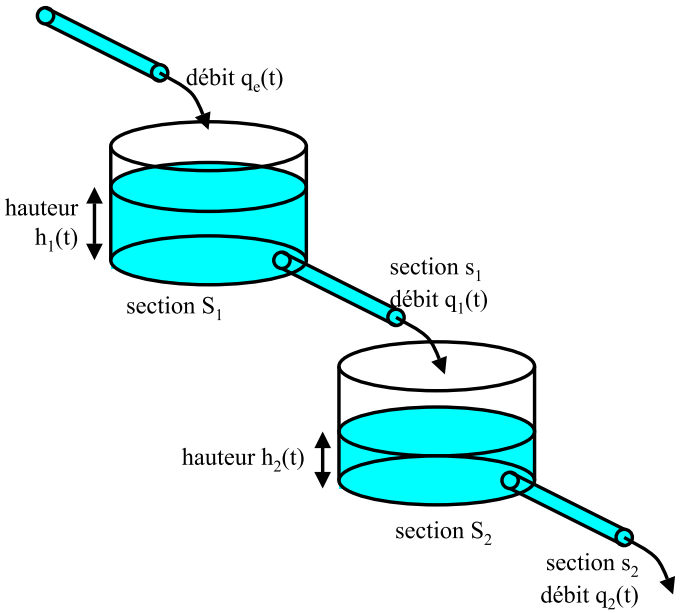
$$S_1 h_1'(t) = q_e(t) - q_1(t)$$
$$S_2 h_2'(t) = q_1(t) - q_2(t)$$

où les débits sont donnés par :
$q_1(t) = s_1\sqrt{2gh_1(t)}$ et
$q_2(t) = s_2\sqrt{2gh_2(t)}$

(b) Linéarisation et simulation du système linéaire

- **Données :** fichier `dataTD32.mat` sur Arche

Système à modéliser



Modélisation

$$S_1 h_1'(t) = q_e(t) - q_1(t)$$
$$S_2 h_2'(t) = q_1(t) - q_2(t)$$

où les débits sont donnés par :
$q_1(t) = s_1\sqrt{2gh_1(t)}$ et
$q_2(t) = s_2\sqrt{2gh_2(t)}$

Pour linéariser on écrit :

$$h_1'(t) = \frac{1}{S_1} q_e(t) - \frac{s_1}{S_1}\sqrt{2gh_1(t)} = f_1(q_e, h_1)$$
$$h_2'(t) = \frac{s_1}{S_2}\sqrt{2gh_1(t)} - \frac{s_2}{S_2}\sqrt{2gh_2(t)} = f_2(h_1, h_2)$$

(b) Linéarisation et simulation du système linéaire

Système non linéaire à linéariser :

$$h_1'(t) = \frac{1}{S_1} q_e(t) - \frac{s_1}{S_1}\sqrt{2gh_1(t)} = f_1(q_e, h_1)$$

$$h_2'(t) = \frac{s_1}{S_2}\sqrt{2gh_1(t)} - \frac{s_2}{S_2}\sqrt{2gh_2(t)} = f_2(h_1, h_2)$$

Déterminer le point d'équilibre imposé par $q_e(t) = q_{e0}$.

(b) Linéarisation et simulation du système linéaire

Système non linéaire à linéariser :

$$h_1'(t) = \frac{1}{S_1} q_e(t) - \frac{s_1}{S_1}\sqrt{2gh_1(t)} = f_1(q_e, h_1)$$
$$h_2'(t) = \frac{s_1}{S_2}\sqrt{2gh_1(t)} - \frac{s_2}{S_2}\sqrt{2gh_2(t)} = f_2(h_1, h_2)$$

Déterminer le point d'équilibre imposé par $q_e(t) = q_{e0}$.

- équilibre $\simeq$ signaux constants :

$$h_1'(t) = 0 \qquad h_1(t) = h_{10}$$
$$h_2'(t) = 0 \qquad h_2(t) = h_{20}$$

(b) Linéarisation et simulation du système linéaire

Système non linéaire à linéariser :

$$h_1'(t) = \frac{1}{S_1} q_e(t) - \frac{s_1}{S_1}\sqrt{2gh_1(t)} = f_1(q_e, h_1)$$
$$h_2'(t) = \frac{s_1}{S_2}\sqrt{2gh_1(t)} - \frac{s_2}{S_2}\sqrt{2gh_2(t)} = f_2(h_1, h_2)$$

Déterminer le point d'équilibre imposé par $q_e(t) = q_{e0}$.

- équilibre $\simeq$ signaux constants :

$$h_1'(t) = 0 \qquad h_1(t) = h_{10}$$
$$h_2'(t) = 0 \qquad h_2(t) = h_{20}$$

- valeurs à l'équilibre :

$$h_{10} = \left(\frac{q_{e0}}{s_1}\right)^2 \frac{1}{2g} = 7.05m$$
$$h_{20} = \left(\frac{q_{e0}}{s_2}\right)^2 \frac{1}{2g} = 5.09m$$

(b) Linéarisation et simulation du système linéaire

Système non linéaire à linéariser :

$$h_1'(t) = \frac{1}{S_1} q_e(t) - \frac{s_1}{S_1}\sqrt{2gh_1(t)} = f_1(q_e, h_1)$$

$$h_2'(t) = \frac{s_1}{S_2}\sqrt{2gh_1(t)} - \frac{s_2}{S_2}\sqrt{2gh_2(t)} = f_2(h_1, h_2)$$

Trouver les équations différentielles linéaires en $\delta q_e(t)$, $\delta h_1(t)$ et $\delta h_2(t)$.

(b) Linéarisation et simulation du système linéaire

Système non linéaire à linéariser :

$$h_1'(t) = \frac{1}{S_1} q_e(t) - \frac{s_1}{S_1}\sqrt{2gh_1(t)} = f_1(q_e, h_1)$$
$$h_2'(t) = \frac{s_1}{S_2}\sqrt{2gh_1(t)} - \frac{s_2}{S_2}\sqrt{2gh_2(t)} = f_2(h_1, h_2)$$

Trouver les équations différentielles linéaires en $\delta q_e(t)$, $\delta h_1(t)$ et $\delta h_2(t)$.

- Le développement de Taylor à l'ordre 1 de f_1 et f_2 :

$$f_1(q_e, h_1) = f_1(q_{e0}, h_{10}) + \left(\frac{\partial f_1}{\partial q_e}\right)_{eq} (q_e(t) - q_{e0}) + \left(\frac{\partial f_1}{\partial h_1}\right)_{eq} (h_1(t) - h_{10})$$
$$f_2(h_1, h_2) = f_2(h_{10}, h_{20}) + \left(\frac{\partial f_2}{\partial h_1}\right)_{eq} (h_1(t) - h_{10}) + \left(\frac{\partial f_2}{\partial h_2}\right)_{eq} (h_2(t) - h_{20})$$

(b) Linéarisation et simulation du système linéaire

Système non linéaire à linéariser :

$$h_1'(t) = \frac{1}{S_1} q_e(t) - \frac{s_1}{S_1}\sqrt{2gh_1(t)} = f_1(q_e, h_1)$$
$$h_2'(t) = \frac{s_1}{S_2}\sqrt{2gh_1(t)} - \frac{s_2}{S_2}\sqrt{2gh_2(t)} = f_2(h_1, h_2)$$

Trouver les équations différentielles linéaires en $\delta q_e(t)$, $\delta h_1(t)$ et $\delta h_2(t)$.

- Le développement de Taylor à l'ordre 1 de f_1 et f_2 :

$$f_1(q_e, h_1) = f_1(q_{e0}, h_{10}) + \left(\frac{\partial f_1}{\partial q_e}\right)_{eq}(q_e(t) - q_{e0}) + \left(\frac{\partial f_1}{\partial h_1}\right)_{eq}(h_1(t) - h_{10})$$
$$f_2(h_1, h_2) = f_2(h_{10}, h_{20}) + \left(\frac{\partial f_2}{\partial h_1}\right)_{eq}(h_1(t) - h_{10}) + \left(\frac{\partial f_2}{\partial h_2}\right)_{eq}(h_2(t) - h_{20})$$

- donne les équations différentielles linéaires :

$$\delta h_1'(t) = \frac{1}{S_1}\delta q_e(t) - \frac{s_1\sqrt{2g}}{2S_1\sqrt{h_{10}}}\delta h_1(t)$$
$$\delta h_2'(t) = \frac{s_1\sqrt{2g}}{2S_2\sqrt{h_{10}}}\delta h_1(t) - \frac{s_2\sqrt{2g}}{2S_2\sqrt{h_{20}}}\delta h_2(t)$$

(b) Linéarisation et simulation du système linéaire

Système linéaire :

$$\delta h_1'(t) = \frac{1}{S_1}\delta q_e(t) - \frac{s_1\sqrt{2g}}{2S_1\sqrt{h_{10}}}\delta h_1(t)$$

$$\delta h_2'(t) = \frac{s_1\sqrt{2g}}{2S_2\sqrt{h_{10}}}\delta h_1(t) - \frac{s_2\sqrt{2g}}{2S_2\sqrt{h_{20}}}\delta h_2(t)$$

Trouver les fonctions de transfert entre $\Delta Q_e(p)$, $\Delta H_1(p)$ et $\Delta H_2(p)$.

(b) Linéarisation et simulation du système linéaire

Système linéaire :

$$\delta h_1'(t) = \frac{1}{S_1}\delta q_e(t) - \frac{s_1\sqrt{2g}}{2S_1\sqrt{h_{10}}}\delta h_1(t)$$

$$\delta h_2'(t) = \frac{s_1\sqrt{2g}}{2S_2\sqrt{h_{10}}}\delta h_1(t) - \frac{s_2\sqrt{2g}}{2S_2\sqrt{h_{20}}}\delta h_2(t)$$

Trouver les fonctions de transfert entre $\Delta Q_e(p)$, $\Delta H_1(p)$ et $\Delta H_2(p)$.

- La propriété $\mathcal{L}(f'(t)) = pF(p) - f(0)$ et $\delta h_i(0) = 0$ donnent :

$$\Delta H_1(p) = \left(\frac{\frac{1}{S_1}}{p + \frac{s_1\sqrt{2g}}{2S_1\sqrt{h_{10}}}} \right) \Delta Q_e(p)$$

$$\Delta H_2(p) = \left(\frac{\frac{s_1\sqrt{2g}}{2S_2\sqrt{h_{10}}}}{p + \frac{s_2\sqrt{2g}}{2S_2\sqrt{h_{20}}}} \right) \Delta H_1(p)$$

(b) Linéarisation et simulation du système linéaire

Système linéaire :

$$\delta h_1'(t) = \frac{1}{S_1}\delta q_e(t) - \frac{s_1\sqrt{2g}}{2S_1\sqrt{h_{10}}}\delta h_1(t)$$

$$\delta h_2'(t) = \frac{s_1\sqrt{2g}}{2S_2\sqrt{h_{10}}}\delta h_1(t) - \frac{s_2\sqrt{2g}}{2S_2\sqrt{h_{20}}}\delta h_2(t)$$

Trouver les fonctions de transfert entre $\Delta Q_e(p)$, $\Delta H_1(p)$ et $\Delta H_2(p)$.

- La propriété $\mathcal{L}(f'(t)) = pF(p) - f(0)$ et $\delta h_i(0) = 0$ donnent :

$$\Delta H_1(p) = \left(\frac{\frac{1}{S_1}}{p + \frac{s_1\sqrt{2g}}{2S_1\sqrt{h_{10}}}} \right) \Delta Q_e(p)$$

$$\Delta H_2(p) = \left(\frac{\frac{s_1\sqrt{2g}}{2S_2\sqrt{h_{10}}}}{p + \frac{s_2\sqrt{2g}}{2S_2\sqrt{h_{20}}}} \right) \Delta H_1(p)$$

Faire les simulations avec les données de `dataTD32.mat` et les fonctions `tf([...],[...])`, `lsim(G1,dqe,t)`, `subplot(...,...,...)`, etc.

(b) Linéarisation et simulation du système linéaire

Faire les simulations avec les données de `dataTD32.mat`.

(b) Linéarisation et simulation du système linéaire

Faire les simulations avec les données de `dataTD32.mat`.

- Calculer le point d'équilibre :
- Définir les fonctions de transfert (fonction `tf([...],[...])`) :
- Simuler les réponses de $G_1(p)$ et $G_2(p)$ (fonction `lsim(G1,...,...)`) :
- Tracer les sorties mesurée et simulée (fonctions `subplot(...,...,...)` et `plot(...,...)`) :

(b) Linéarisation et simulation du système linéaire

Faire les simulations avec les données de `dataTD32.mat`.

- Calculer le point d'équilibre :

```
h10=(1/(2*g))*(qe0/s1)^2;
h20=(1/(2*g))*(qe0/s2)^2;
```

- Définir les fonctions de transfert (fonction `tf([...],[...])`) :

- Simuler les réponses de $G_1(p)$ et $G_2(p)$ (fonction `lsim(G1,...,...)`) :

- Tracer les sorties mesurée et simulée (fonctions `subplot(...,...,...)` et `plot(...,...)`) :

(b) Linéarisation et simulation du système linéaire

Faire les simulations avec les données de `dataTD32.mat`.

- Calculer le point d'équilibre :

```
h10=(1/(2*g))*(qe0/s1)^2;
h20=(1/(2*g))*(qe0/s2)^2;
```

- Définir les fonctions de transfert (fonction `tf([...],[...])`) :

```
G1=tf([1],[S1 s1*sqrt(g)/sqrt(2*h10)]);
G2=tf([s1*sqrt(g)/sqrt(2*h10)],[S2 s2*sqrt(g)/sqrt(2*h20)]);
```

- Simuler les réponses de $G_1(p)$ et $G_2(p)$ (fonction `lsim(G1,...,...)`) :

- Tracer les sorties mesurée et simulée (fonctions `subplot(...,...,...)` et `plot(...,...)`) :

(b) Linéarisation et simulation du système linéaire

Faire les simulations avec les données de `dataTD32.mat`.

- Calculer le point d'équilibre :

```
h10=(1/(2*g))*(qe0/s1)^2;
h20=(1/(2*g))*(qe0/s2)^2;
```

- Définir les fonctions de transfert (fonction `tf([...],[...])`) :

```
G1=tf([1],[S1 s1*sqrt(g)/sqrt(2*h10)]);
G2=tf([s1*sqrt(g)/sqrt(2*h10)],[S2 s2*sqrt(g)/sqrt(2*h20)]);
```

- Simuler les réponses de $G_1(p)$ et $G_2(p)$ (fonction `lsim(G1,...,...)`) :

```
dqe=qe-qe0;
t=[0:1:24*3600];
dh1=lsim(G1,dqe,t);
dh2=lsim(G2,dh1,t);
h1=dh1+h10;
h2=dh2+h20;
```

- Tracer les sorties mesurée et simulée (fonctions `subplot(...,...,...)` et `plot(...,...)`) :

(b) Linéarisation et simulation du système linéaire

Faire les simulations avec les données de `dataTD32.mat`.

- Calculer le point d'équilibre :

```
h10=(1/(2*g))*(qe0/s1)^2;
h20=(1/(2*g))*(qe0/s2)^2;
```

- Définir les fonctions de transfert (fonction `tf([...],[...])`) :

```
G1=tf([1],[S1 s1*sqrt(g)/sqrt(2*h10)]);
G2=tf([s1*sqrt(g)/sqrt(2*h10)],[S2 s2*sqrt(g)/sqrt(2*h20)]);
```

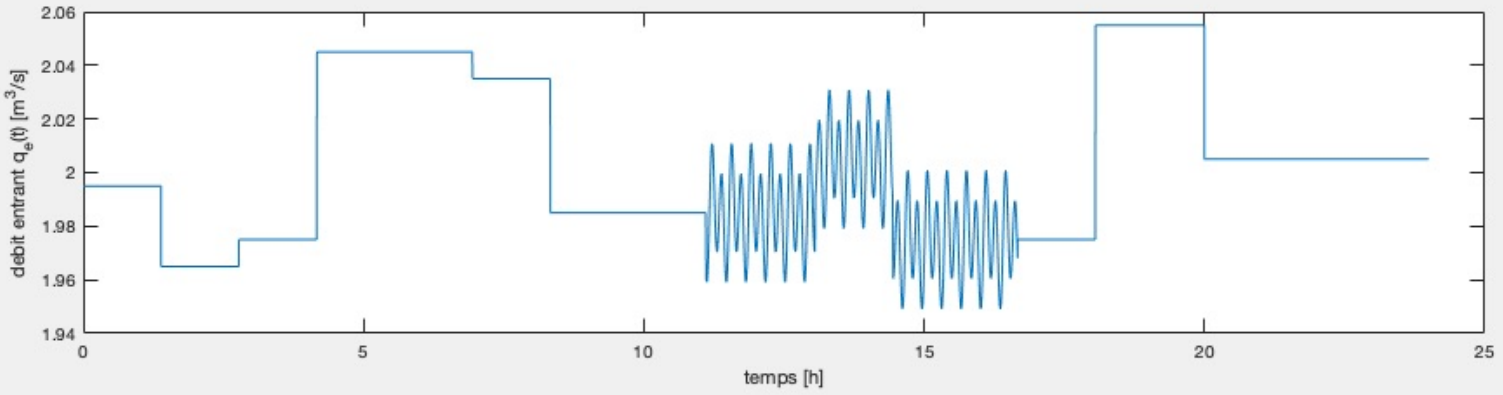
- Simuler les réponses de $G_1(p)$ et $G_2(p)$ (fonction `lsim(G1,...,...)`) :

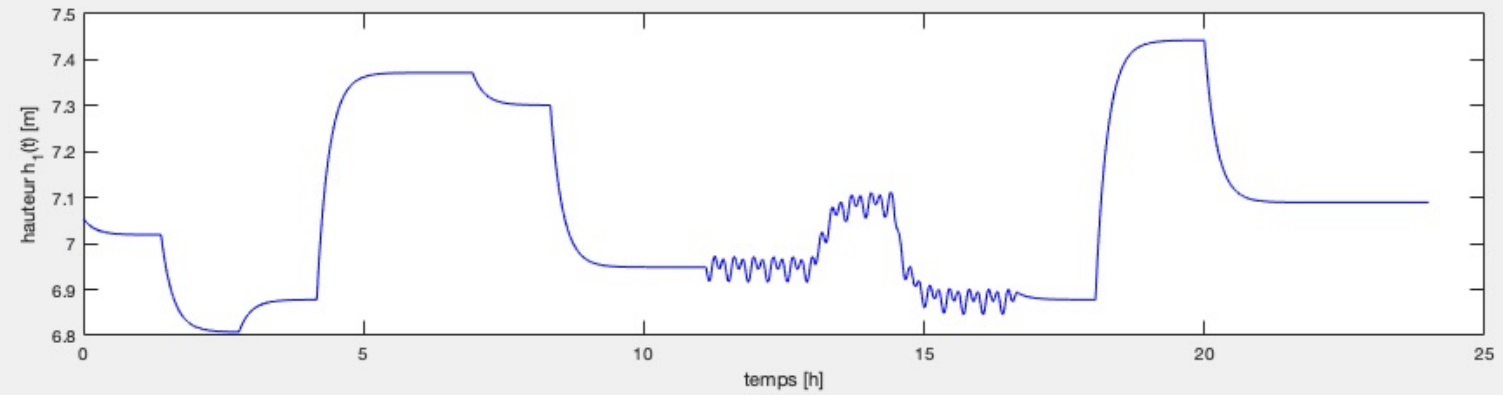
```
dqe=qe-qe0;
t=[0:1:24*3600];
dh1=lsim(G1,dqe,t);
dh2=lsim(G2,dh1,t);
h1=dh1+h10;
h2=dh2+h20;
```

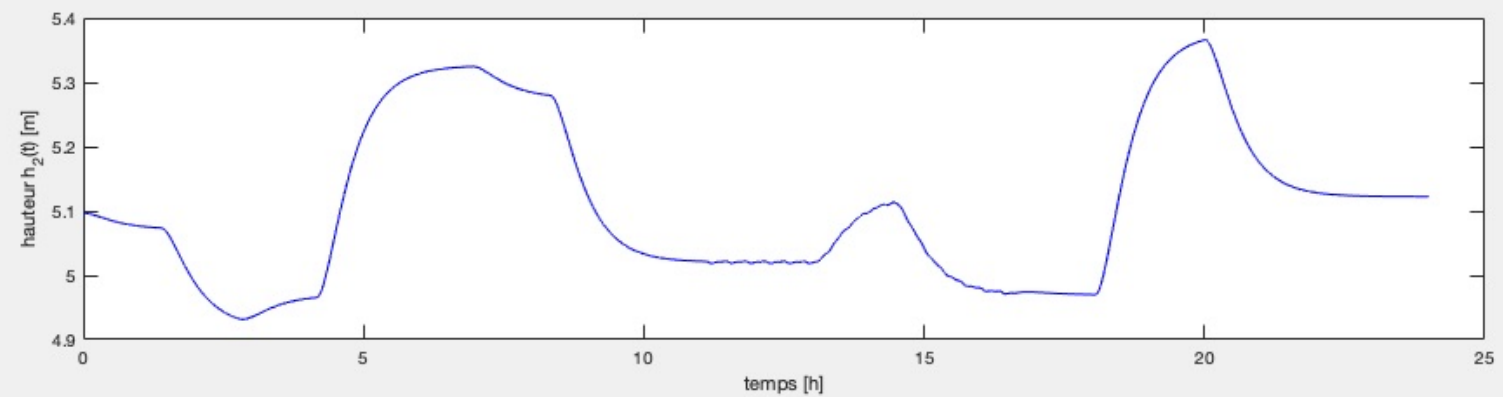
- Tracer les sorties mesurée et simulée (fonctions `subplot(...,...,...)` et `plot(...,...)`) :

```
figure
subplot(3,1,1), plot(t/3600,qe)
subplot(3,1,2), plot(t/3600,h1)
subplot(3,1,3), plot(t/3600,h2)
```

(b) Linéarisation et simulation du système linéaire







(b) Simulation du système non linéaire

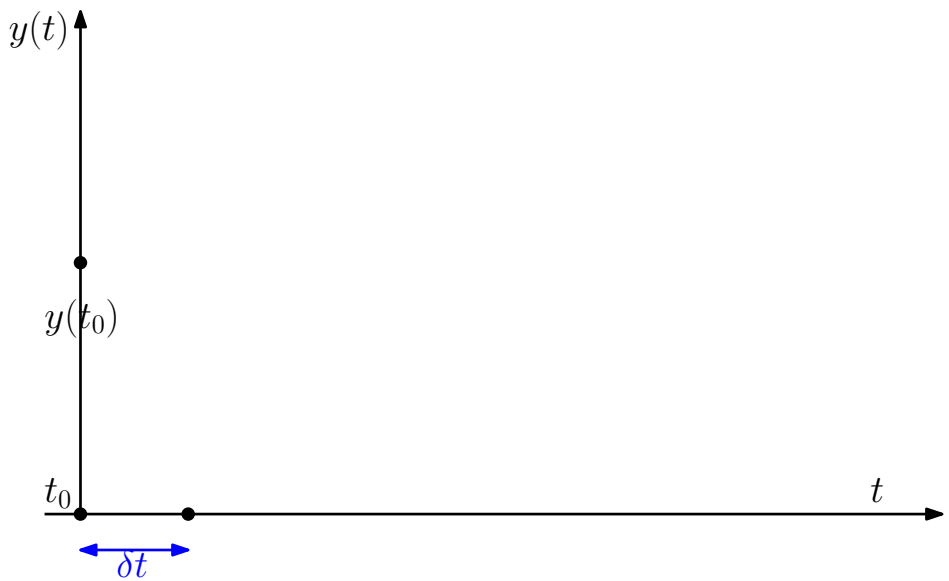
Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor

(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

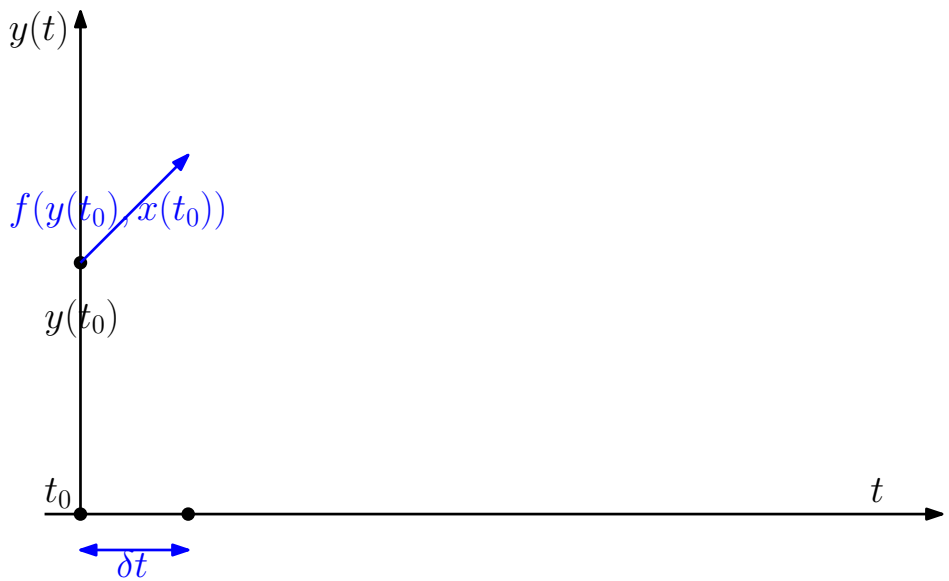
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

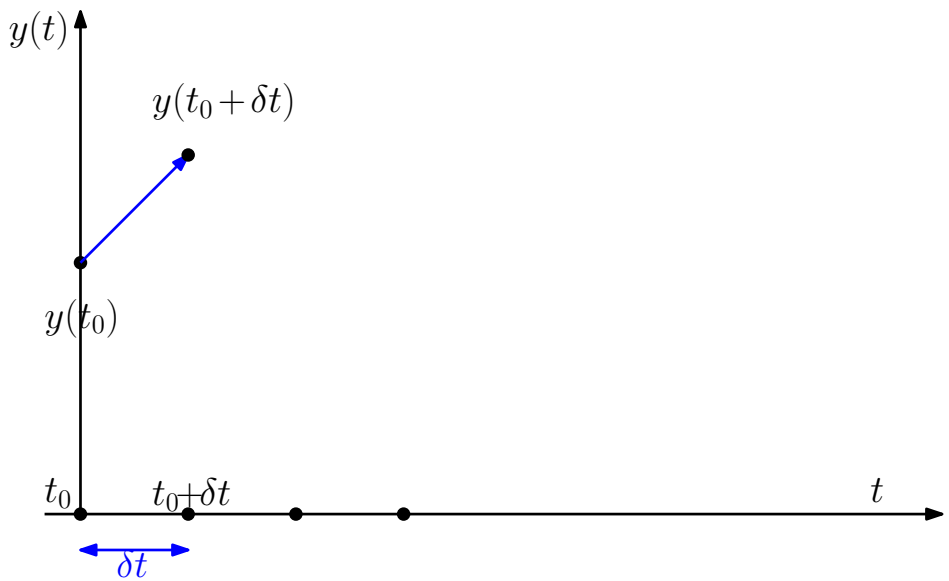
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

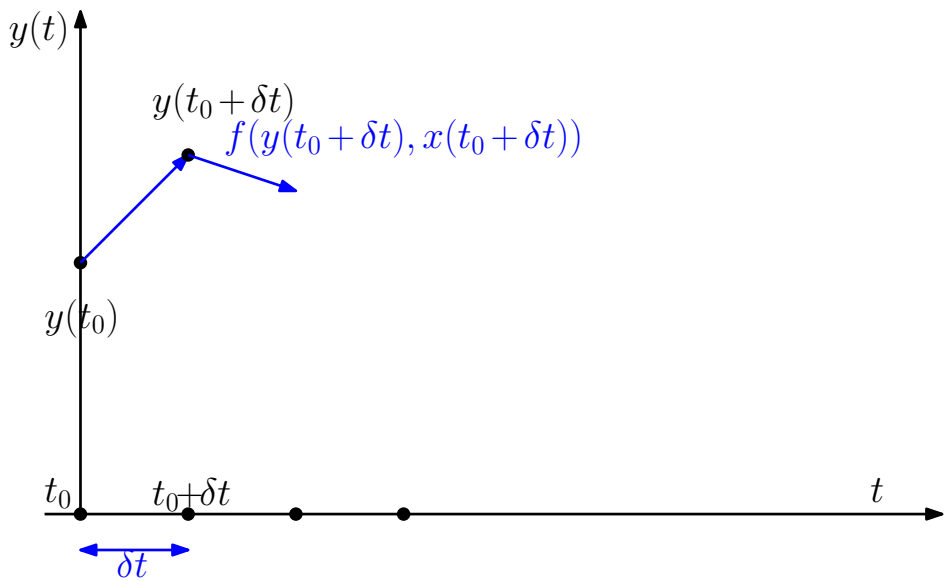
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

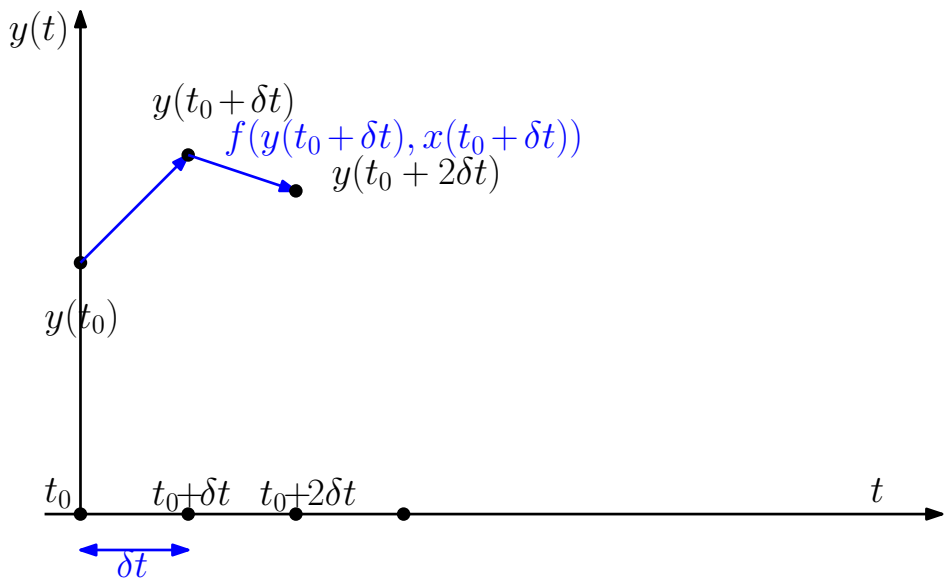
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$
- en $t_0 + \delta t$: $y(t_0 + 2\delta t) = y(t_0 + \delta t) + f(y(t_0 + \delta t), x(t_0 + \delta t))\delta t$



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

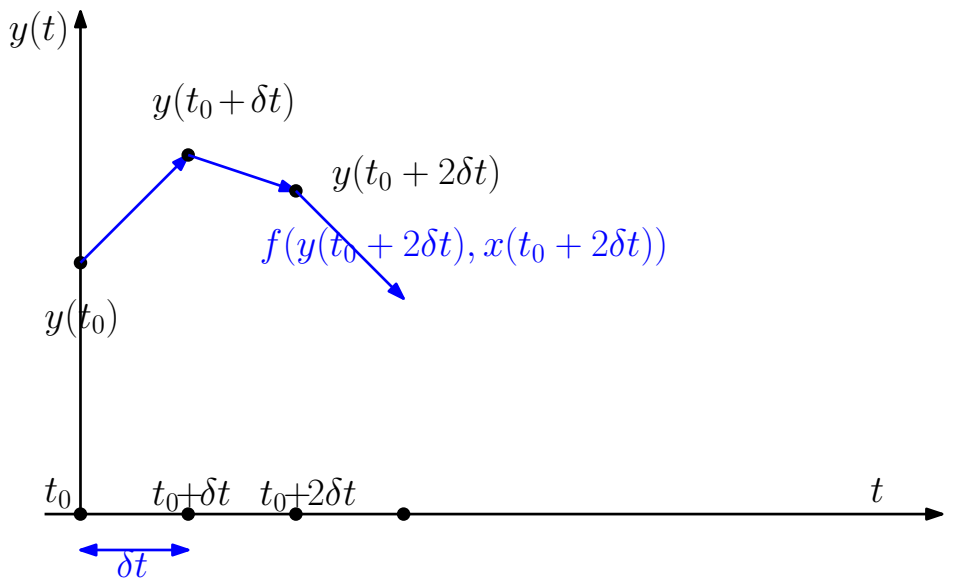
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$
- en $t_0 + \delta t$: $y(t_0 + 2\delta t) = y(t_0 + \delta t) + f(y(t_0 + \delta t), x(t_0 + \delta t))\delta t$



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

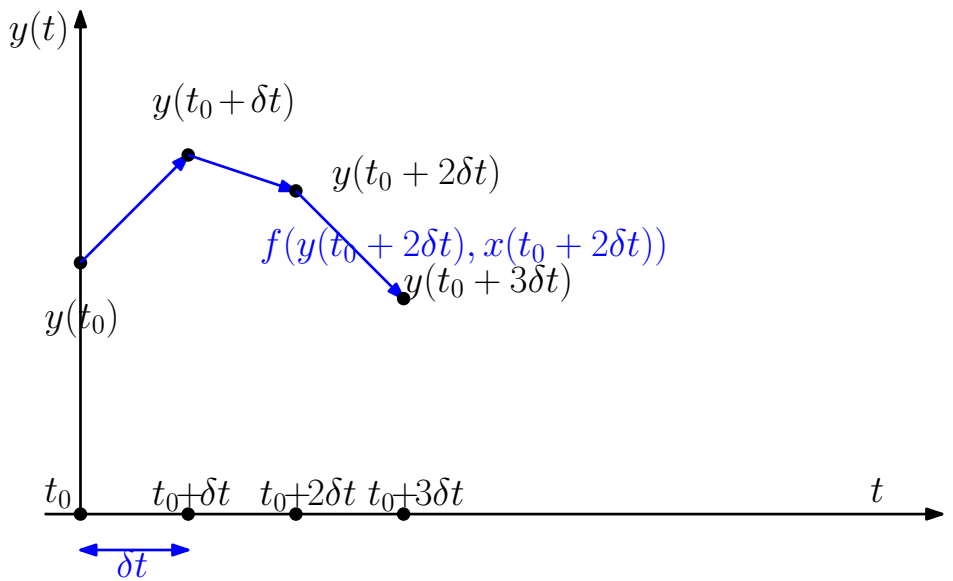
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$
- en $t_0 + \delta t$: $y(t_0 + 2\delta t) = y(t_0 + \delta t) + f(y(t_0 + \delta t), x(t_0 + \delta t))\delta t$
- en $t_0 + 2\delta t$: $y(t_0 + 3\delta t) = y(t_0 + 2\delta t) + f(y(t_0 + 2\delta t), x(t_0 + 2\delta t))\delta t$



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

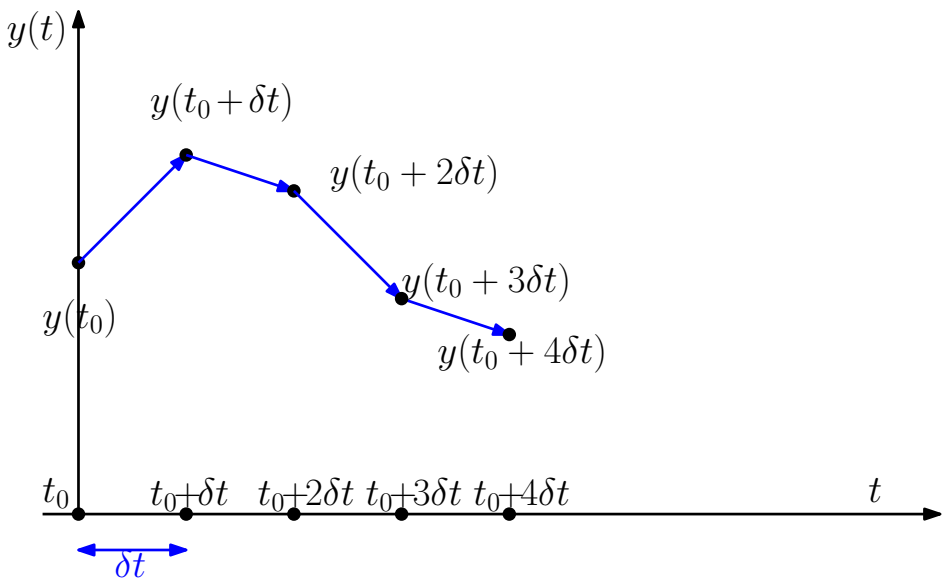
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$
- en $t_0 + \delta t$: $y(t_0 + 2\delta t) = y(t_0 + \delta t) + f(y(t_0 + \delta t), x(t_0 + \delta t))\delta t$
- en $t_0 + 2\delta t$: $y(t_0 + 3\delta t) = y(t_0 + 2\delta t) + f(y(t_0 + 2\delta t), x(t_0 + 2\delta t))\delta t$



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

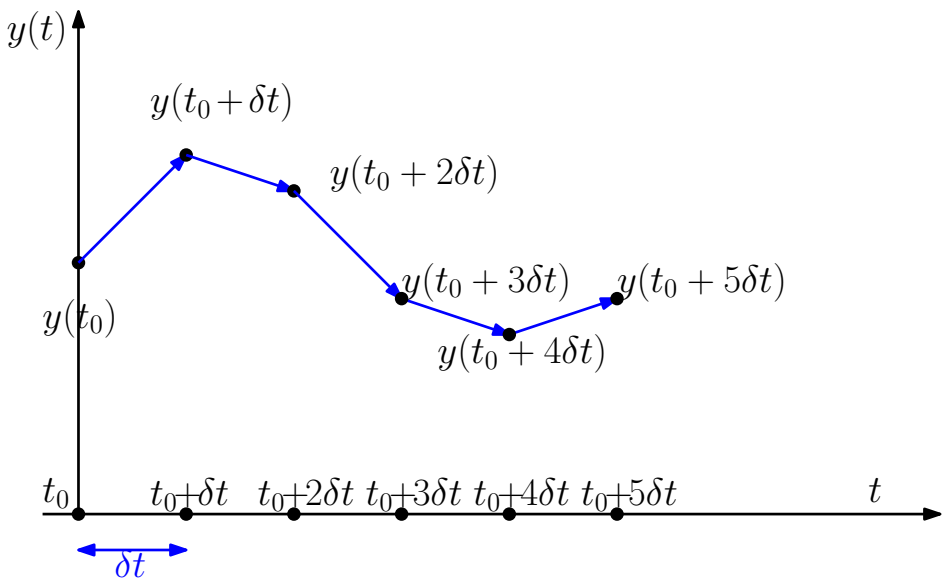
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$
- en $t_0 + \delta t$: $y(t_0 + 2\delta t) = y(t_0 + \delta t) + f(y(t_0 + \delta t), x(t_0 + \delta t))\delta t$
- en $t_0 + 2\delta t$: $y(t_0 + 3\delta t) = y(t_0 + 2\delta t) + f(y(t_0 + 2\delta t), x(t_0 + 2\delta t))\delta t$
- ...



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

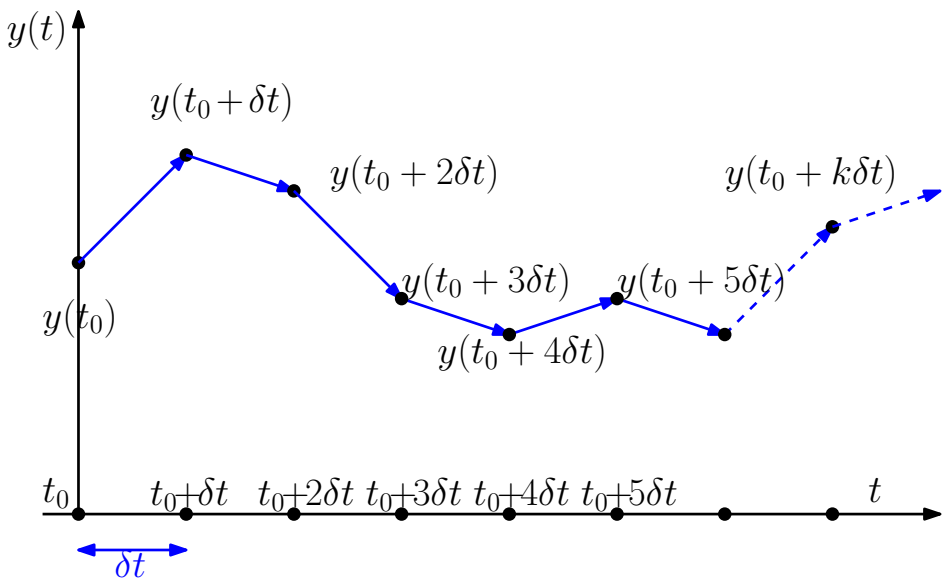
- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$
- en $t_0 + \delta t$: $y(t_0 + 2\delta t) = y(t_0 + \delta t) + f(y(t_0 + \delta t), x(t_0 + \delta t))\delta t$
- en $t_0 + 2\delta t$: $y(t_0 + 3\delta t) = y(t_0 + 2\delta t) + f(y(t_0 + 2\delta t), x(t_0 + 2\delta t))\delta t$
- ...



(b) Simulation du système non linéaire

Intégration numérique de $y'(t) = f(y(t), x(t))$ pour $y(t_0)$ et $x(t)$ connus.

- solution numérique $\{y(t_0 + k\delta t)\}_{k=1,\ldots,N}$ avec des dev^t de Taylor
- en t_0 : $y(t_0 + \delta t) = y(t_0) + f(y(t_0), x(t_0))\delta t$
- en $t_0 + \delta t$: $y(t_0 + 2\delta t) = y(t_0 + \delta t) + f(y(t_0 + \delta t), x(t_0 + \delta t))\delta t$
- en $t_0 + 2\delta t$: $y(t_0 + 3\delta t) = y(t_0 + 2\delta t) + f(y(t_0 + 2\delta t), x(t_0 + 2\delta t))\delta t$
- ...
- en $t_0 + k\delta t$: $y(t_0 + (k+1)\delta t) = y(t_0 + k\delta t) + f(y(t_0 + k\delta t), x(t_0 + k\delta t))\delta t$



(b) Simulation du système non linéaire

On utilise la fonction `ode23` pour l'intégration numérique à δt variable

- Voir l'aide de `ode23` pour son utilisation (`help ode23`)

(b) Simulation du système non linéaire

On utilise la fonction `ode23` pour l'intégration numérique à δt variable

- Voir l'aide de `ode23` pour son utilisation (`help ode23`)

(1) Dans l'éditeur, créer une fonction qui calcule h' à partir de t, h (et $+$ si affinité)

(2) Dans la fenêtre principale, utiliser `ode23`

(3) Comparer les résultats des simulations des systèmes linéaire et non linéaire.

(b) Simulation du système non linéaire

On utilise la fonction `ode23` pour l'intégration numérique à δt variable

- Voir l'aide de `ode23` pour son utilisation (`help ode23`)

(1) Dans l'éditeur, créer une fonction qui calcule h' à partir de t, h (et $+$ si affinité)

```
function [dh]=resnl(t,h,S1,S2,s1,s2,qe)
qet=interp1([0:24*3600],qe,t);
dh(1)= (qet-s1*sqrt(2*9.81*h(1)))/S1;
dh(2)= (s1*sqrt(2*9.81*h(1))-s2*sqrt(2*9.81*h(2)))/S2;
dh=[dh(1) ; dh(2)];
```

(2) Dans la fenêtre principale, utiliser `ode23`

(3) Comparer les résultats des simulations des systèmes linéaire et non linéaire.

(b) Simulation du système non linéaire

On utilise la fonction `ode23` pour l'intégration numérique à δt variable

- Voir l'aide de ode23 pour son utilisation (`help ode23`)

(1) Dans l'éditeur, créer une fonction qui calcule h' à partir de t, h (et + si affinité)

```
function [dh]=resnl(t,h,S1,S2,s1,s2,qe)
qet=interp1([0:24*3600],qe,t);
dh(1)= (qet-s1*sqrt(2*9.81*h(1)))/S1;
dh(2)= (s1*sqrt(2*9.81*h(1))-s2*sqrt(2*9.81*h(2)))/S2;
dh=[dh(1) ; dh(2)];
```

(2) Dans la fenêtre principale, utiliser ode23

```
[tnl,hnl]=ode23(@resnl,[0:24*3600],[h10;h20],[],S1,S2,s1,s2,qe)
```

(3) Comparer les résultats des simulations des systèmes linéaire et non linéaire.

(b) Simulation du système non linéaire

On utilise la fonction `ode23` pour l'intégration numérique à δt variable

- Voir l'aide de ode23 pour son utilisation (`help ode23`)

(1) Dans l'éditeur, créer une fonction qui calcule h' à partir de t, h (et + si affinité)

```
function [dh]=resnl(t,h,S1,S2,s1,s2,qe)
qet=interp1([0:24*3600],qe,t);
dh(1)= (qet-s1*sqrt(2*9.81*h(1)))/S1;
dh(2)= (s1*sqrt(2*9.81*h(1))-s2*sqrt(2*9.81*h(2)))/S2;
dh=[dh(1) ; dh(2)];
```

(2) Dans la fenêtre principale, utiliser ode23

```
[tnl,hnl]=ode23(@resnl,[0:24*3600],[h10;h20],[],S1,S2,s1,s2,qe)
```

(3) Comparer les résultats des simulations des systèmes linéaire et non linéaire.

```
figure
subplot(3,1,1), plot(t/3600,qe)
subplot(3,1,2), plot(t/3600,h1,tn/3600,hnl(:,1))
subplot(3,1,3), plot(t/3600,h2,tnl/3600,hnl(:,2))
```

(b) Simulation du système non linéaire et comparaison