Machine Asynchrone - 3

v4

Préambule

Le but de cet exercice est d'utiliser Matlab pour calculer des caractéristiques de couple.

Au final cet exercice est une simple application de l'équation du couple et de l'équivalent de Thévenin, dans différents cas, afin de comprendre comment appliquer ces équations et aussi visualiser toutes les différentes caractéristiques de couple vues au cours.

Donnée

Une machine asynchrone a les caractéristiques suivantes :

- $U_n = 230$ V : tension de phase nominale
- $I_n = 5$ A : courant de phase nominal
- $f_n = 50$ Hz : fréquence nominale
- $p = 2$: nombre de paires de pôles
- $r_s = 0.02$ pu : résistance statorique en p.u.
- $x_{\sigma s} = 0.12$ pu : réactance de fuite statorique en p.u.
- $x_h = 2$ pu : réactance de champ principal en p.u.
- $r'_r = 0.03$ pu : résistance rotorique rapportée au stator en p.u.
- $x'_{\sigma r} = 0.12$ pu : réactance de fuite rotorique rapportée au stator en p.u.

Calcul des vraies grandeurs

L'impédance de phase nominale vaut :

$$Z_n = \frac{U_n}{I_n} = 46\,[\Omega] \tag{1}$$

Ainsi, les paramètres en vraies grandeurs valent :

$$R_s = r_s\, Z_n = 0.92\,[\Omega] \tag{2}$$
$$X_{\sigma s} = x_{\sigma s}\, Z_n = 5.52\,[\Omega] \tag{3}$$
$$X_h = x_h\, Z_n = 92\,[\Omega] \tag{4}$$
$$R'_r = r'_r\, Z_n = 1.38\,[\Omega] \tag{5}$$
$$X'_{\sigma r} = x'_{\sigma r}\, Z_n = 5.52\,[\Omega] \tag{6}$$

Ecriture des fonctions pour l'équivalent de Thévenin et pour le couple électromagnétique

Equivalent de Thévenin

Les 2 équations utiles à l'établissement de l'équivalent de Thévnin sont :

$$\underline{U}_e = \underline{U}_s \frac{jX_h}{R_s + j(X_{\sigma s} + X_h)} \tag{7}$$

$$\underline{Z}_e = jX_h \frac{R_s + jX_{\sigma s}}{R_s + j(X_{\sigma s} + X_h)} = R_e + jX_e \tag{8}$$

Ainsi la fonction reçoit en entrée:

- U_s : la tension de phase statorique (valeur complexe mais choisie avec un angle 0 donc équivalent à la norme)
- R_s : la résistance statorique
- $X_{\sigma s}$: la réactance de fuite statorique
- X_h : la réactance de champ principal

et donne en sortie

- U_e : la tension équivalente de Thévenin (la norme suffit pour le calcul du couple)
- R_e : la partie réelle de l'impédance équivalente de Thévenin
- X_e : la partie imaginaire de l'impédance équivalente de Thévenin

Le code Matlab correspondant est donc :

```
function [Ue, Re, Xe] = eq_thevenin(Us, Rs, Xss, Xh)
    Ue_cmpx = Us*1i.*Xh./(Rs+1i*(Xss+Xh));
    Ue = abs(Ue_cmpx);
    Ze_cmpx = 1i*Xh.*(Rs+1i*Xss)./(Rs+1i*(Xss+Xh));
    Re = real(Ze_cmpx);
    Xe = imag(Ze_cmpx);
end
```

La notation . (*element wise*) est autant utilisée pour la mutliplication avec U_s que pour les divisions faisant intervenir les réactances ($X_{\sigma s}$ et X_h), car nous passerons à cette fonction des vecteurs où U_s et f (donc X) pourront varier.

Couple électromagnétique

L'équation du couple électromagnétique est :

$$T_{em} = \frac{3U_e^2 \frac{R'_r}{s}}{\Omega_s \left[\left(R_e + \frac{R'_r}{s} \right)^2 + (X_e + X'_{\sigma r})^2 \right]} \tag{9}$$

Ainsi la fonction reçoit en entrée:

- s : le glissement
- U_s : la tension de phase statorique (valeur complexe mais choisie avec un angle 0 donc équivalent à la norme)
- Ωs : la vitesse angulaire en [rad/s] dans le monde mécanique
- R_s : la résistance statorique
- $X_{\sigma s}$: la réactance de fuite statorique
- X_h : la réactance de champ principal
- R'_r : la résistance rotorique rapportée au stator
- $X'_{\sigma r}$: la réactance de fuite rotorique rapportée au stator

et donne en sortie

- T_{em} : le couple électromagnétique

Le code Matlab correspondant est donné ci-dessous. On commence par appeler la fonction de l'équivalent de Thévenin définie précédemment afin d'avoir U_e, R_e, X_e et ensuite on écrit l'équation du couple électromagnétique. Attention à écrire l'équation avec les "." nécessaires afin de pouvoir passer le glissement comme un vecteur de tous les glissements et obtenir en sortie un vecteur de tous les couples correspondants.

```
function [Tem] = torque(s, Us, Omegas, Rs, Xss, Xh, Xsrp, Rrp)
    [Ue, Re, Xe] = eq_thevenin(Us, Rs, Xss, Xh);
    Tem = 3*Ue.^2*Rrp./s./(Omegas.*((Re+Rrp./s).^2+(Xe+Xsrp).^2));
end
```

La notation . (*element wise*) est autant utilisée pour les *carrés* ^{2} que pour s, afin de permettre de générer le couple en 1 seul calcul ayant autant U_s que s sous forme de vecteur, ce qui nous permettra aisément de faire varier U_s et s.

Résolution

1. Conditions nominales

La machine est alimentée à *tension* et *fréquence* nominales, en mode moteur et en mode génératrice.

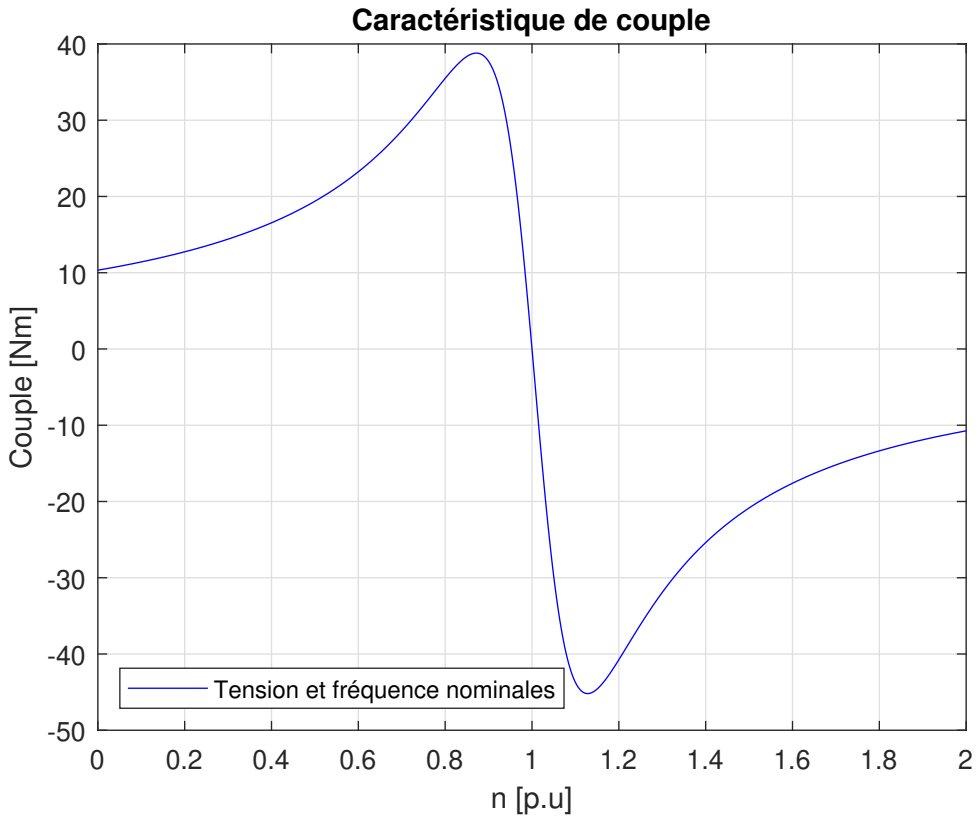
Comme nous étudions le mode moteur et le mode génératrice, le glissement va de -1 à 1.

Après, il suffit d'appeler la fonction du couple électromagnétique avec les bons paramètres.

```
s = linspace(-1,1,2000);
Tem = torque(s, Un, Omegas, Rs, Xss, Xh, Xsrp, Rrp);
```

Pour les axes, l'axe x est en fonction de la vitesse n en $[pu]$ et vaut donc $1 - s$.

```
x_axis = 1-s; % n [pu]
y_axis = Tem;
```



2. Tension variable

Nous étudions maintenant uniquement le mode moteur.

On ajoute une sécurité pour le cas $s = 0$, et nous remplaçons la valeur par un nombre très petit afin de ne pas avoir une erreur de division par 0.

```
s = linspace(0,1,1000);
s(s==0)= 1e-6;
```

Et nous allons maintenant appliquer une tension variable selon :

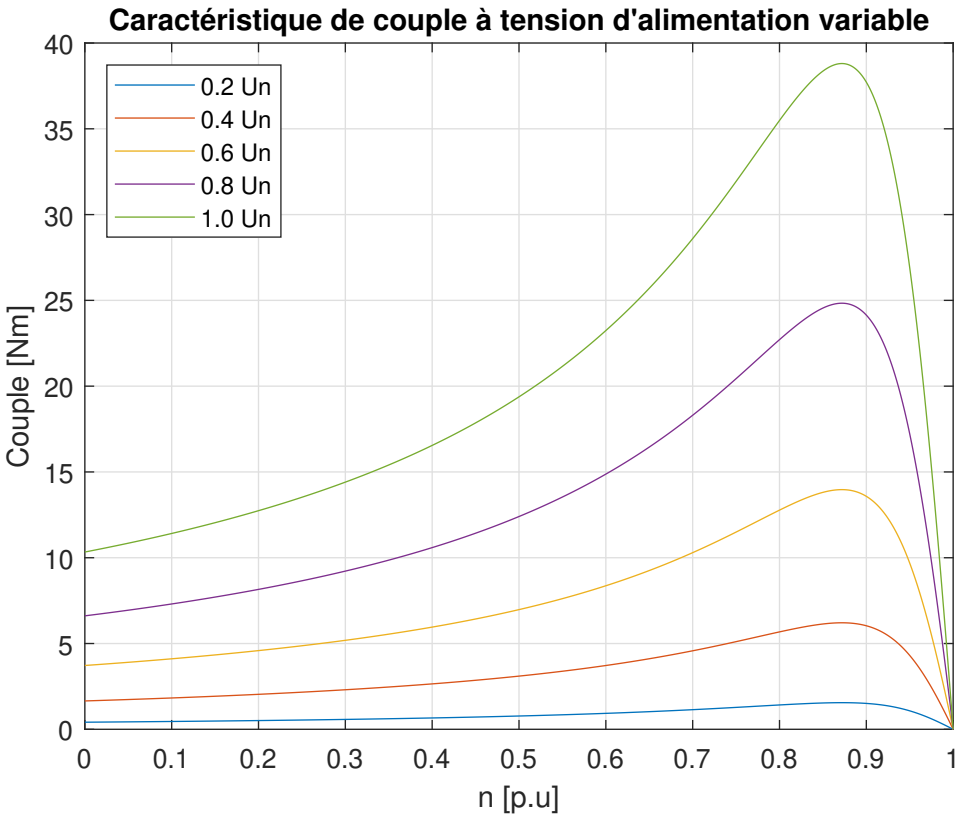
```
s = linspace(0,1,1000);
Us = [0.2; 0.4; 0.6; 0.8; 1]*Un;
```

Il suffit ensuite d'appeler la fonction du couple électromagnétique avec les bons paramètres.

```
Tem_Uvar = torque(s, Us, Omegas, Rs, Xss, Xh, Xsrp, Rrp);
```

Pour les axes, l'axe x est en fonction de la vitesse n en $[pu]$ et vaut donc $1 - s$.

```
x_axis = 1-s; % n [pu]
y_axis = Tem_Uvar;
```



3. U et f variables (USA)

Toujours pour le mode moteur, donc s a déjà été défini au point précédent On recalcule le couple en europe pour l'avoir en référence pour la comparaison.

```
Tem = torque(s, Un, Omegas, Rs, Xss, Xh, Xsrp, Rrp);
```

Aux USA, la tension et la fréquence ne sont pas les même.

```
U_USA = 110; % [V]
f_USA = 60;  % [A]
Omegas_USA = 2*pi*f_USA/p %[rad/s]
```

Ainsi nous pouvons définir le rapport entre la fréquence aux USA et notre fréquence de référence pour adapter les valeurs des réactances lors de l'utilisation aux USA.

En effet, une réactance est définie par

$$X = \omega L \tag{10}$$

où ω est la pulsation dans le monde électrique. Il faut donc adapter les réactances en proportion.

Le code Matlab est donc le suivant :

```
ratio_USA = f_USA/fn;
Xss_USA = ratio_USA*Xss     %[Ohm]
Xhs_USA = ratio_USA*Xh      %[Ohm]
Xsrp_USA = ratio_USA*Xsrp   %[Ohm]
```

et de là il suffit d'appeler la fonction du calcule du couple électromagnétique avec les bons paramètres :

```
Tem_USA = torque(s, U_USA, Omegas_USA, Rs, Xss_USA, ...
    Xhs_USA, Xsrp_USA, Rrp);
```

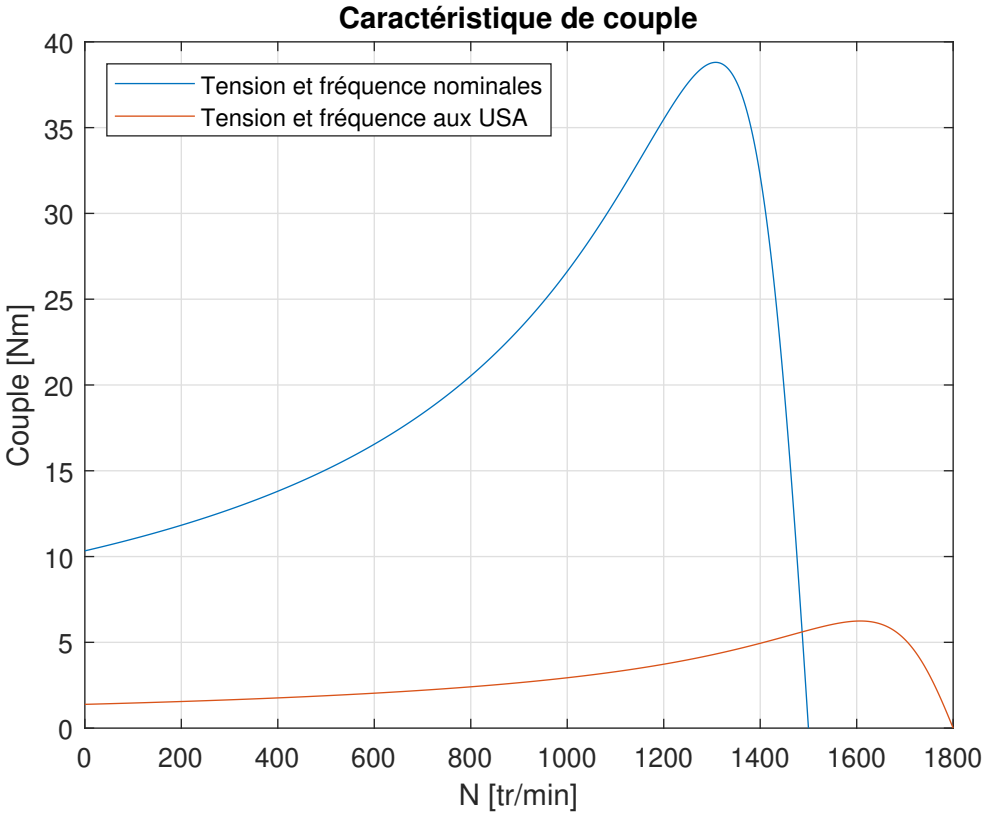
Pour les axes, l'axe x est en fonction de la vitesse N en $[tr/min]$ et vaut donc :

$$N = (1 - s)\, 60\, f/p \tag{11}$$

où s est le glissement, f la fréquence et p le nombre de paires de poles.

Dans notre cas, le code Matlab correspondant est :

```
x_axis = (1-s)*60*fn/p; % N [tr/min]
y_axis = Tem;
x_axis_USA = (1-s)*60*f_USA/p; % N [tr/min]
y_axis_USA = Tem_USA;
```



4. Us/f constant

Toujours pour le mode moteur, nous allons maintenant étudier l'alimentation lorsque le rapport Us/f est conservé constant selon :

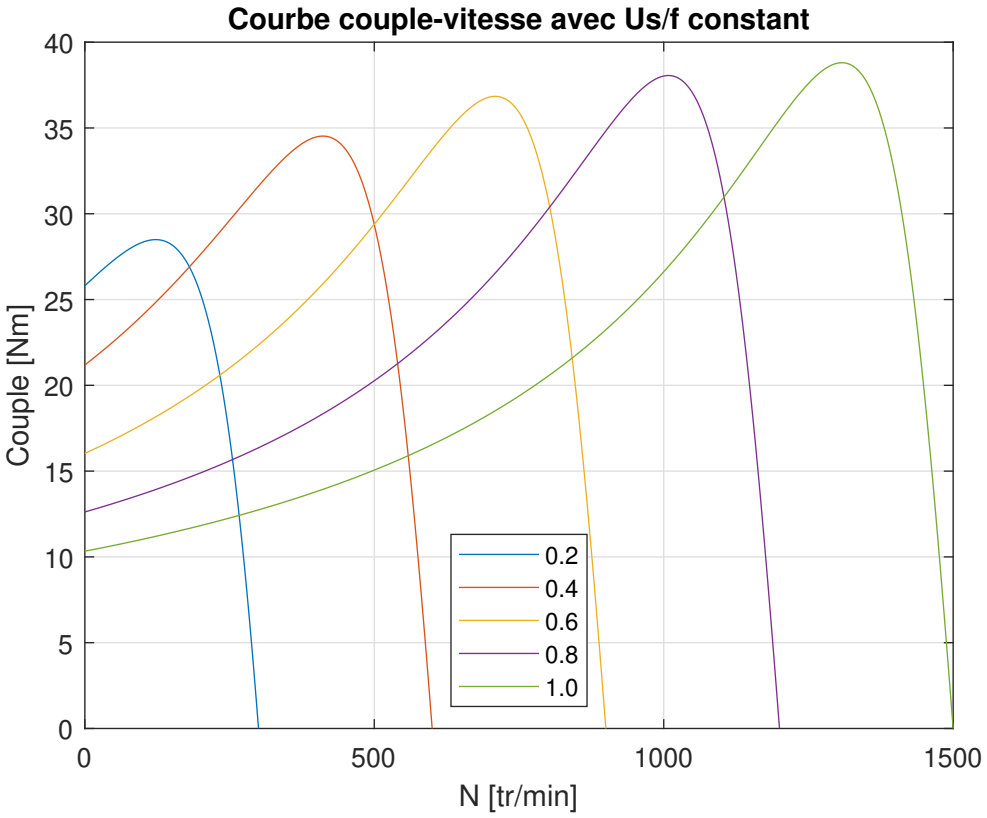
```
ratio = [0.2; 0.4; 0.6; 0.8; 1];
```

Ce *ratio* doit maintenant être appliqué autant à U_n, qu'à Ω_s, qu'aux réactances. Ainsi, l'appel à la fonction prend la forme :

```
Tem_Us_sur_f = torque(s, ratio*Un, ratio*Omegas, Rs, ratio*Xss, ...
    ratio*Xh, ratio*Xsrp, Rrp);
```

Pour les axes, l'axe x est en fonction de la vitesse N en $[tr/min]$ et a été défini à l'équation (11) et le code Matlab correspond dans ce cas vaut :

```
x_axis = (1-s)*60*fn.*ratio/p; % N [tr/min]
y_axis = Tem_Us_sur_f;
```



5. Rhéostat

Toujours pour le mode moteur, nous allons maintenant étudier une machine asynchrone à rotor bobiné connectée à un Rhéostat valant un certain nombre de fois R'_r :

```
Rheostat = [1; 2; 4; 9];
```

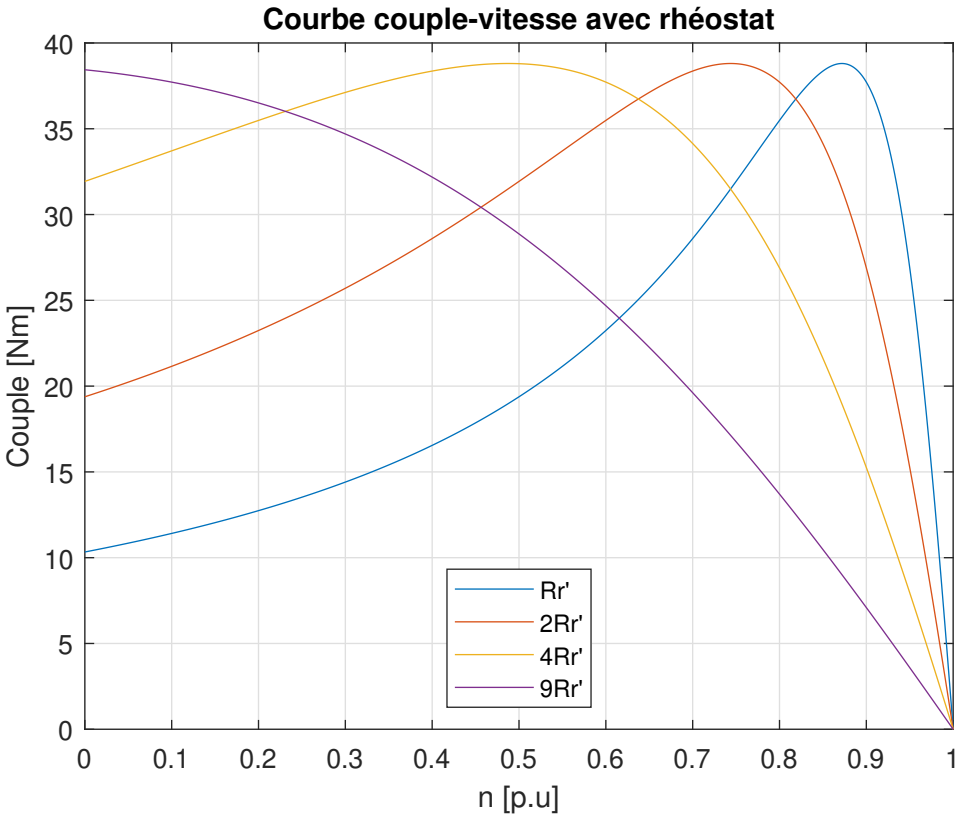
Ces valeurs de *Rheostat* doivent maintenant être multipliées à R'_r.

Ainsi, l'appel à la fonction prend la forme :

```
Tem_Rh = torque(s, Un, Omegas, Rs, Xss, Xh, Xsrp, Rheostat*Rrp);
```

Pour les axes, l'axe x est en fonction de la vitesse n en $[pu]$ et vaut donc $1 - s$.

```
x_axis = 1-s; % n [pu]
y_axis = Tem_Rh;
```



Calcul des courants à l'instant du démarrage

Nous avons déjà étudié cela dans l'exercice MAS-02, une partie du code peut donc être copié/collé et sera ensuite ajusté à l'utilisation d'un rhéostat.

En utilisant un rhéostat, l'expression de l'impédance équivalente prend la forme :

$$\underline{Z}_{eq} = R_s + jX_{\sigma s} + \frac{jX_h\left(\frac{R'_r+R'_{rh}}{s} + jX'_{\sigma r}\right)}{\frac{R'_r+R'_{rh}}{s} + j\left(X_h + X'_{\sigma r}\right)} \tag{12}$$

Dans notre cas, les valeurs de $R'_r + R'_{rh}$ vont être prises comme un nombre entier de fois R'_r.

Le code Matlab est donc le suivant, en rappelant qu'à l'instant du démarrage $s = 1$:

```
s= 1;

% Impedance complexe
Zeq_cmpx_start = Rs+1i*Xss+(1i*Xh*(((Rheostat.*Rrp)/s)+1i*Xsrp))./ ...
    (((Rheostat.*Rrp)/s)+1i*(Xh+Xsrp)); % [Ohm]

% Norme de l'impedance complexe
Zeq_start = abs(Zeq_cmpx_start); % [Ohm]

% Courant au demarrage
Istart = Un./Zeq_start; % [A]

% Courant au demarrage en p.u.
istart = Istart/In % [pu]
```

Comme attendu, les valeurs numériques obtenues montrent que le courant de démarrage diminue avec l'augmentation du rhéostat.