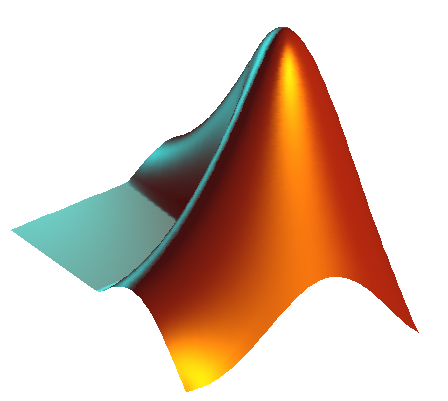
|  |
| --- |
|  |
| Modélisation multi physique |
| Essai de modélisation de machine thermodynamique |

|  |
| --- |
| Jacques Pichon |

Sommaire

[Essai de modélisation multi-physique d’un ballon thermodynamique 2](#_Toc422166429)

[Avant-propos 2](#_Toc422166430)

[1 Modélisation 2](#_Toc422166431)

[1.1 Le modèle de départ 2](#_Toc422166432)

[1.2 L’adaptation pour le ballon thermodynamique 3](#_Toc422166433)

[1.3 Les résultats 5](#_Toc422166434)

[2 Acquisition 7](#_Toc422166435)

[2.1 Le GUI 7](#_Toc422166436)

[2.2 Bilan de ce dispositif de mesure : 8](#_Toc422166437)

[2.3 Analyse de l’écart entre la mesure et la simulation 9](#_Toc422166438)

[3 Limite du modèle : 9](#_Toc422166439)

[Annexes : 10](#_Toc422166440)

[1 Script Matlab pilotage.m 10](#_Toc422166441)

[2 Les capteurs 13](#_Toc422166442)

[2.1 Capteur de température : Danfoss AKS-11 13](#_Toc422166443)

[2.2 Capteur de pression : Danfoss AKS 32R 14](#_Toc422166444)

Essai de modélisation multi-physique d’un ballon thermodynamique

# Avant-propos

Depuis la version 2014a de Matlab, une démonstration d’utilisation des circuits thermo fluidiques est proposée dans les exemples de Simscape. L’étude d’un réfrigérateur permet d’observer le comportement du fluide dans différents points du circuit. C’est à partir de cette démo que j’ai tenté d’adapter le modèle en pompe à chaleur pour un ballon thermodynamique.

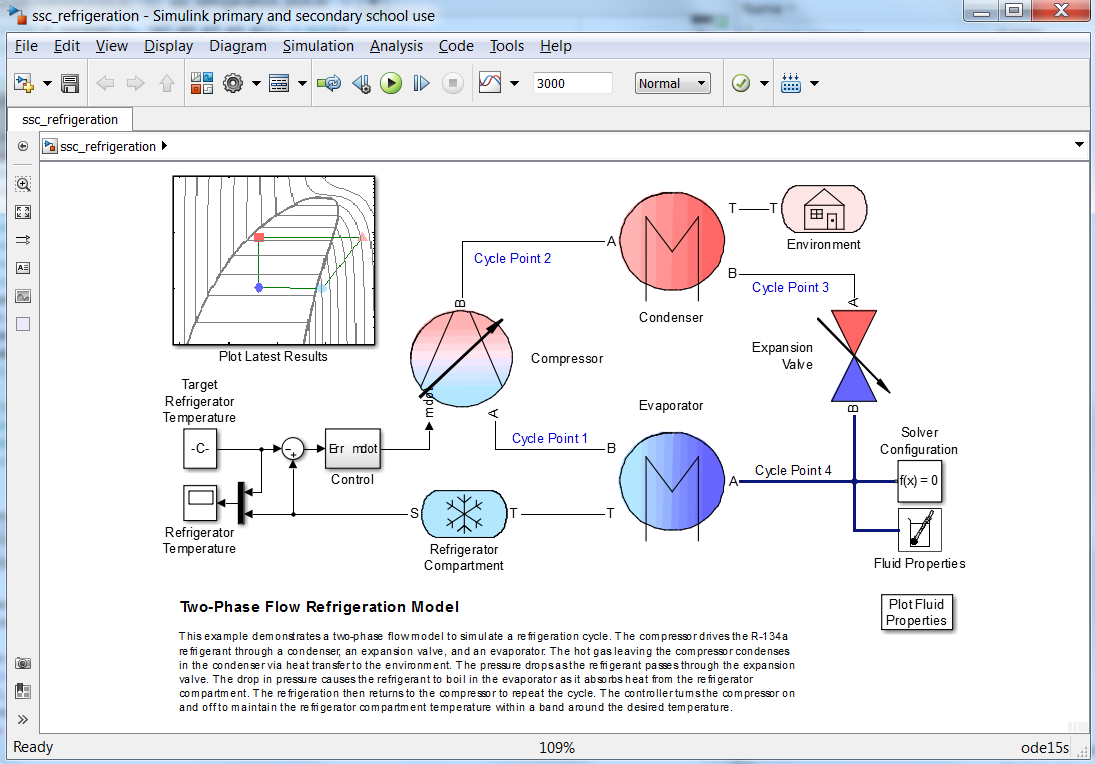
La montée en température du ballon est observable de façon fiable. Les différentes informations sur le fluide sont affichées en fin de simulation, avec un diagramme de Mollier animé.

# Modélisation

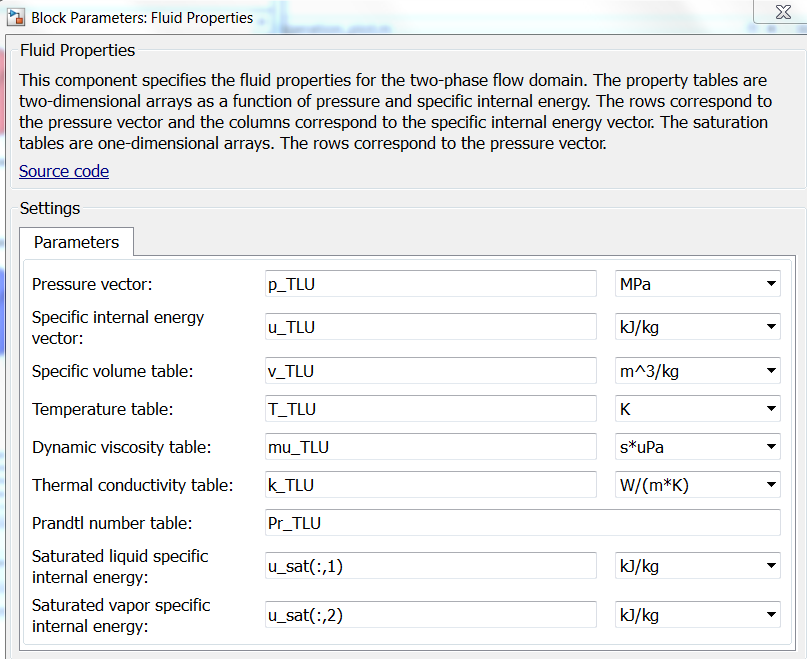
## Le modèle de départ

L’étude des machines thermo fluidiques est maintenant disponible avec l’intégration des propriétés des fluides dans les circuits. Un exemple d’utilisation sur un réfrigérateur est proposé dans les exemples de Simscape.



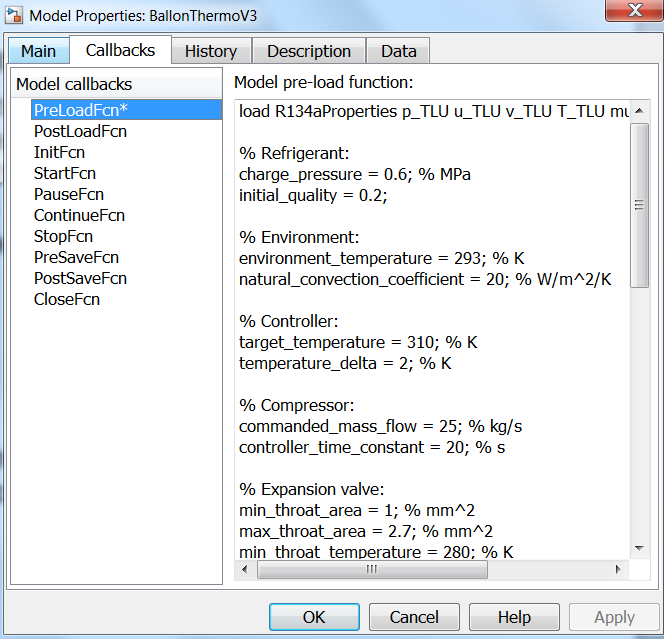


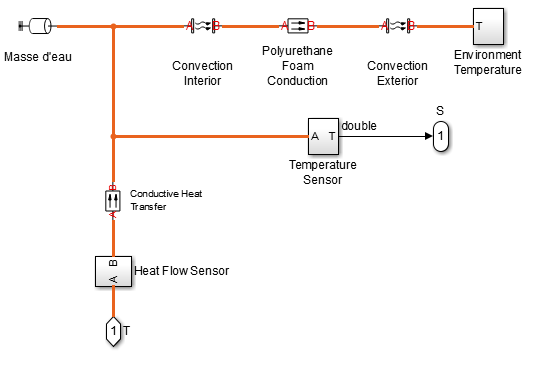
Les caractéristiques du fluide, R-134a, sont enrtegistrées dans un tableau de valeurs.



Plusieurs résultats sont enregistrés pendant la simulation pour observer l’évolution des paramètres. La montée en température du chauffe-eau est observée dans un scope.

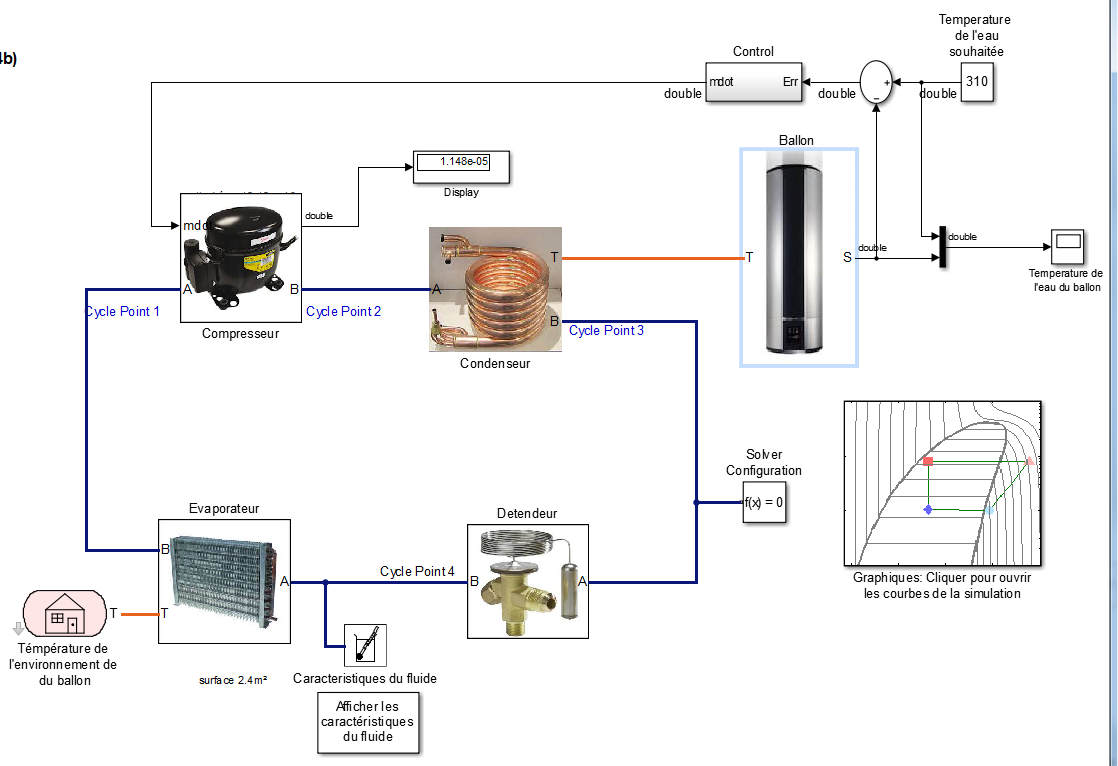
## L’adaptation pour le ballon thermodynamique

Pour notre étude du ballon thermodynamique, plusieurs adaptations du modèle exemple ont été nécessaires. De nombreux paramètres sur les caractéristiques des composants restent incertains, les surfaces des échangeurs, les caractéristiques de la vanne de détente, etc… Ces paramètres pour l’étude d’un équipement spécifique seront à rechercher et à personnaliser dans les variables du modèle. Ce réglage est à modifier dans les propriétés du modèle dans la fonction « Pré\_Loadfcn ».



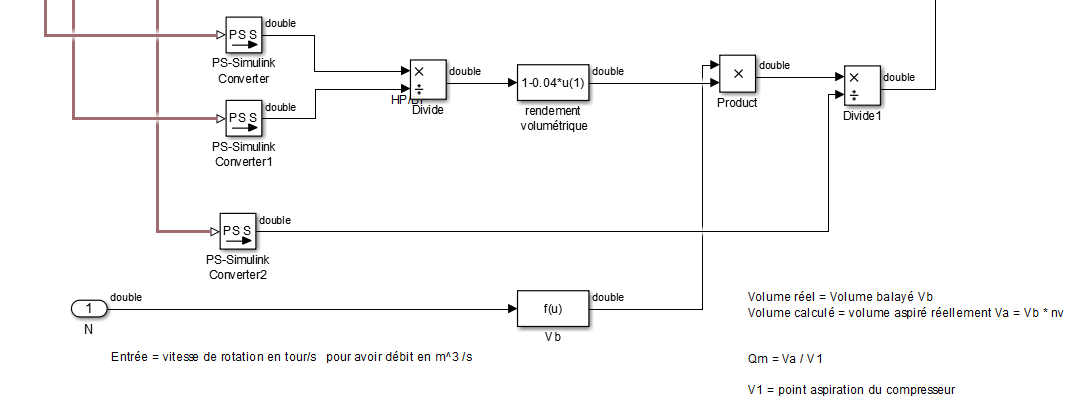
Dans cette modélisation, le système thermodynamique capte la chaleur pour l’apporter au ballon d’eau chaude sanitaire.

La masse d’eau à chauffer va également perdre des calories avec le milieu environnant.



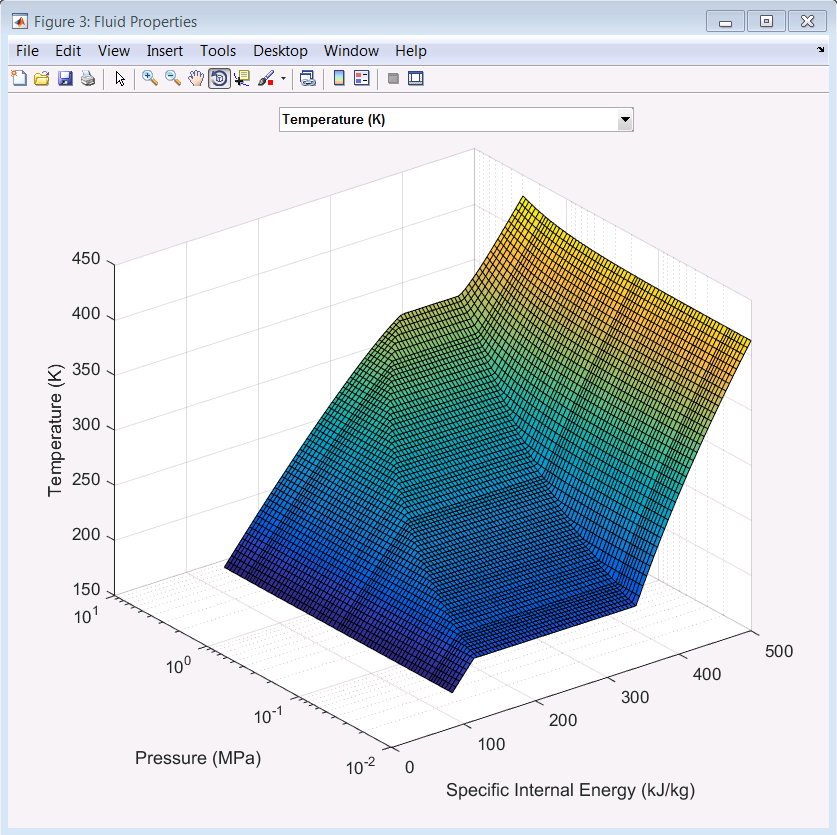
* Pilotage du Compresseur

Le pilotage du compresseur est fait en intégrant le rendement volumétrique, de façon à être proche des pratiques professionnelles du domaine des frigoristes.



Le pilotage est dont fait en vitesse de rotation du moteur du compresseur.

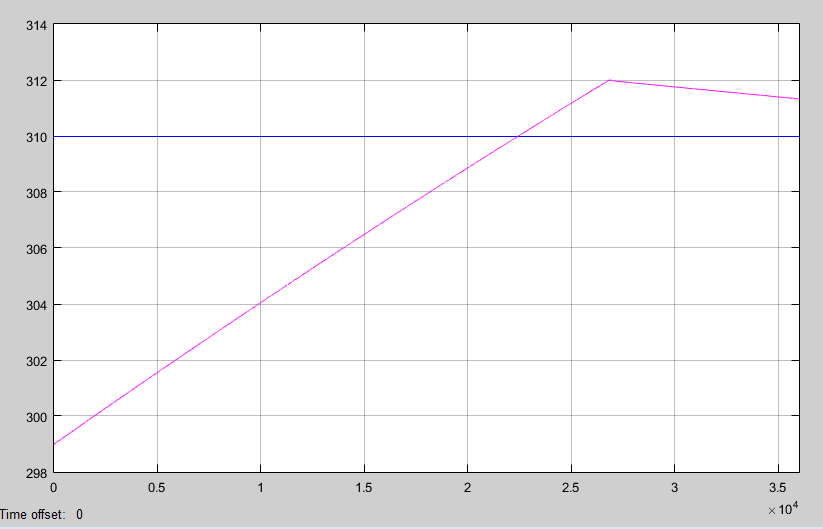
Le fluide utilisé est du R134a.



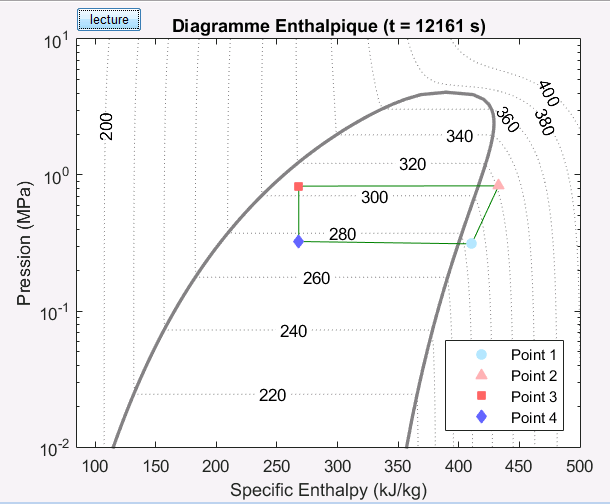
Visualisation des caractéristiques du fluide (tableau de valeurs pour le R134-a).

## Les résultats

Avec les paramètres sélectionnés, nous obtenons la simulation de montée en température suivante.

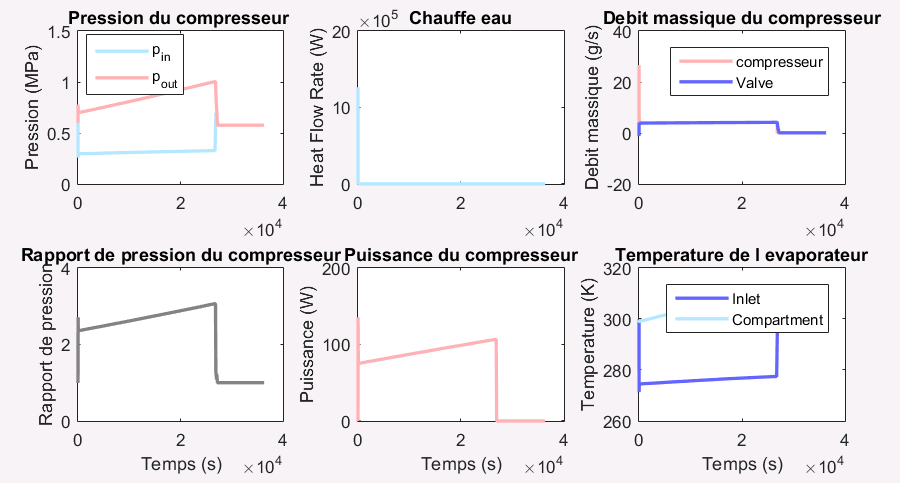


Nous pouvons également observer l’animation du diagramme de Mollier en fonction du temps. Le graphe dynamique est généré à la fin de la simulation.



Les quatre points de mesure sont issus des capteurs placés dans le modèle.

Plusieurs autres grandeurs physiques sont mises en forme pour observer les performances de la machine.

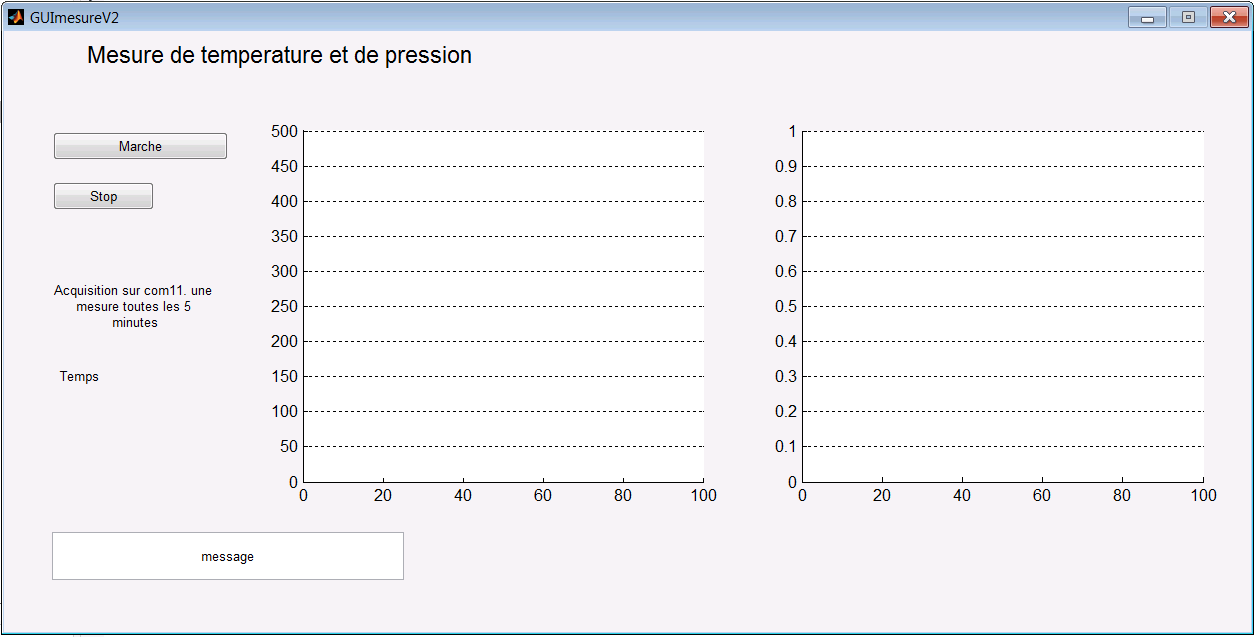


# Acquisition

Pour la mesure des paramètres dans le circuit, nous utilisons une carte Arduino Uno connectée à l’ordinateur. Les capteurs sont pilotés par un script Matlab.

L’acquisition des températures et pressions dans le circuit est faite à intervalle régulier.

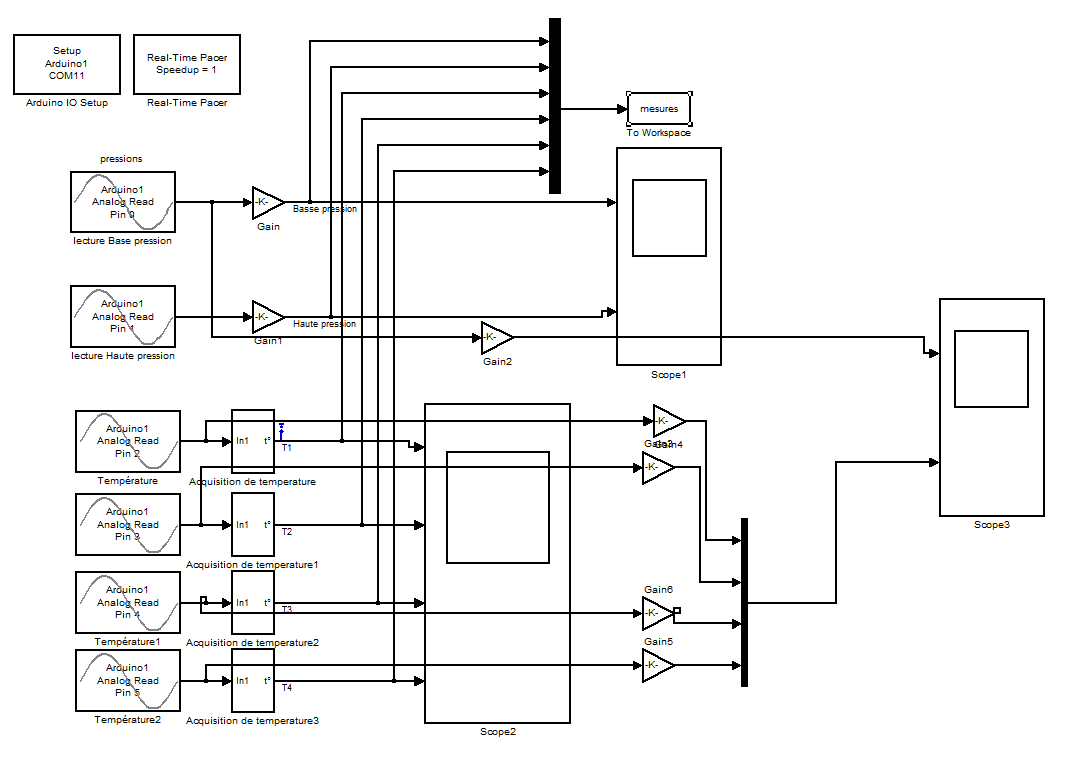
## Le GUI



Le pilotage de l’acquisition est fait par une interface graphique permettant le lancement des mesures. Les points sont positionnés à chaque acquisition, toutes les cinq minutes. Le programme associé au GUI est un script matlab. L’acquisition est faite par le script « pilotage.m », exécuté tant que l’utilisateur n’arrête pas le traitement par une action sur le bouton « stop ».

Le programme enregistre les données de température et de pression dans les points stratégiques du circuit fluidique, les deux pressions, haute et basse et les quatre températures à la sortie des composants. Le fichier est de type CSV pour être importé dans un tableur ou directement dans matlab pour être analysé.

Après une action de l’utilisateur sur le bouton marche, une boucle réalise une acquisition en lançant une simulation du modèle de mesure réalisé avec la bibliothèque Arduino IO. (matlab R2012).



Les scopes sont utilisés uniquement pour la mise au point du modèle. L’ensemble des acquisitions est stocké dans la variable « mesure » dans le workspace de matlab, sous la forme d’une matrice. L’exploitation des valeurs à chaque mesure est la lecture de la première ligne de la matrice. C’est avec cette ligne de valeur que les graphiques sont actualisés et mis à l’échelle.

Les différentes fonctions sont commentées dans le code matlab en annexe.

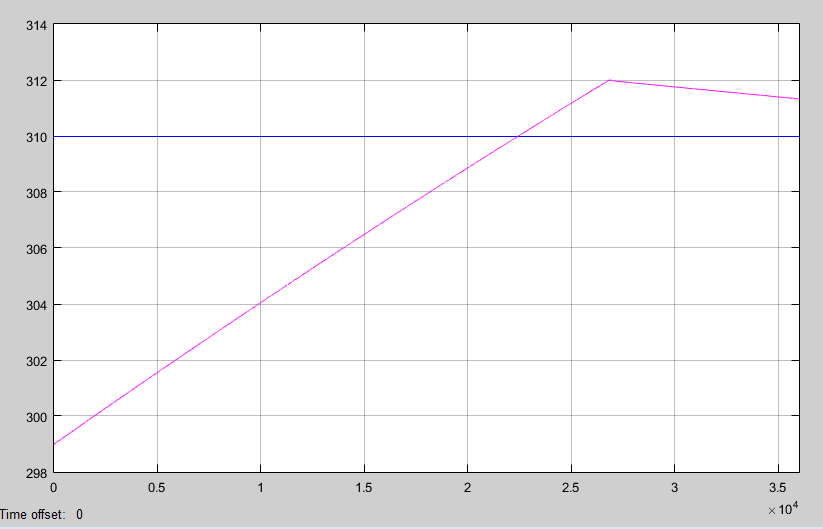
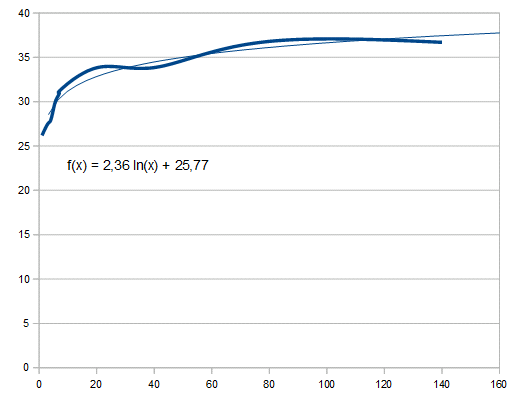
Les capteurs utilisés sont analogiques. Les données sont converties en valeurs directement exploitables.

## Bilan de ce dispositif de mesure :

Avec une carte à bas prix, il est possible d’instrumenter un dispositif pour une longue période d’enregistrement avec un monitoring en temps réel par l’actualisation des courbes sur le PC. La sauvegarde des données dans un fichier permet de mener une analyse des écarts avec une simulation dans une seconde phase de l’étude.

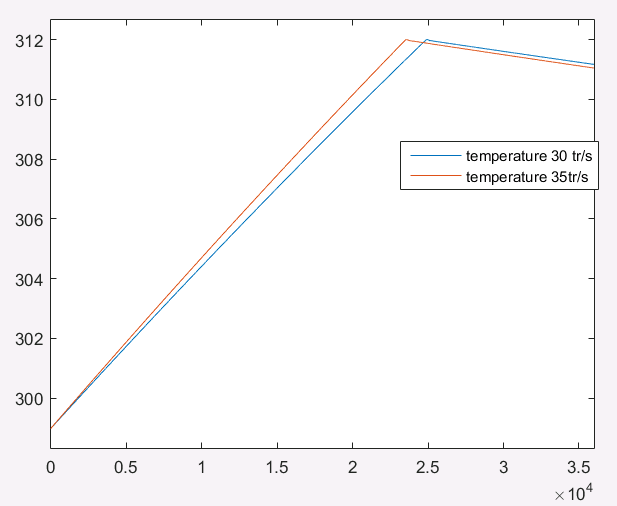
Le modèle utilisé peut être adapté à d’autres systèmes. Les morceaux de script sont réutilisables. Il faut rester vigilent sur la vitesse de traitement de la chaine d’acquisition. La liaison série par le port USB ne permettra pas une acquisition fiable sur des systèmes électroniques rapides.

## Analyse de l’écart entre la mesure et la simulation

Montée en témpérature mesurée Montée en température simulée

La simulation est bien trop lente avec une durée d’environ 5 heures pour monter de 26° à 35° alors que la mesure sur le ballon réel est d’environ une heure. Le paramétrage du modèle nécessite un réglage complexe par la multiplicité des paramètres de l’installation. L’allure de la courbe de température simulée est trop linéaire.



L’intérêt pédagogique de la simulation résidera dans l’observation de l’influence de certains paramètres comme la masse d’eau à chauffer ou l’influence du débit de fluide dans le compresseur par exemple.

La visualisation du diagramme de Mollier animé est très parlante pour observer la baisse de performance du système en fonction de la température de l’eau du ballon.

Influence du débit massique sur la montée en température

# Limite du modèle :

Certains paramètres de l’installation à modéliser sont inconnus. Les valeurs sont restées inchangées depuis l’exemple de départ. La courbe de montée en température est donc bien loin des mesures effectuées sur le ballon étudié.

La lenteur de la montée en température du modèle est certainement due à un débit massique beaucoup trop faible. En fonctionnement normal, le compresseur devrait débiter dans les 40 g.s-1 .Ici, le débit est 10 fois inférieur. Je ne suis pas parvenu à faire fonctionner le modèle avec un pilotage à 48 tour.s-1, vitesse à laquelle le moteur du compresseur devrait normalement tourner. La simulation ne fonctionne plus avec une commande supérieure à 35 tr.s-1.

Ce modèle très complexe sera rendu plus simple dans les futures versions de Simulink puisque les équipes de développement de Matworks travaillent sur une bibliothèque pour le domaine thermofluidique qui sera intégrée dans les outils de Simscape.

Annexes :

# Script Matlab pilotage.m

%open up datalogging file

clock\_now = clock; %gets a 6-element vector of the current date and time in decimal form [year month day hour minute seconds]

clock\_now(6) = round(clock\_now(6)); %round to the nearest second

date\_and\_time\_string = ''; %initialize

for i = 1:1:6 %build up the time into a string

date\_and\_time\_string = [date\_and\_time\_string,num2str(clock\_now(i),'%02d')];

if i==3 || i==5 %after the day, and after the minute, add an underscore in the file name

date\_and\_time\_string = [date\_and\_time\_string,'\_']; %add an underscore (\_)

end

end

%create the "data" directory in the current location, if necessary

if exist('data')~=7 %if "data" does not yet exist

mkdir('data'); %creat it

end

file\_name = ['data/data\_',date\_and\_time\_string,'.csv'];

fid = fopen(file\_name,'w');

%reception des mesures du système

%lancement du temps

tic;

global keep\_going;

keep\_going = true;

loop\_i = 1; %loop counter (which loop # are we on?)

%format du graphe pour la pression : pression

%dans le GUI : graphique

valeurMaxiPression=5;

ylim(handles.pression,[0,valeurMaxiPression])

title(handles.pression,'Mesure des pressions')

xlabel(handles.pression,'temps')

ylabel(handles.pression,'pression en bar')

grid(handles.pression,'on')

%format du graphe pour les températues : graphique

%dans le GUI : temperature

valeurMaxiTemperature=5;

ylim(handles.temperature,[0,valeurMaxiTemperature])

title(handles.temperature,'Mesure des températures')

xlabel(handles.temperature,'temps')

ylabel(handles.temperature,'températures en °c')

grid(handles.temperature,'on')

%preparation des graphiques

num\_itns2plot = 100; %number of iterations to plot.

plot\_handles = NaN(num\_itns2plot,7); %create an array of NaNs

handle\_i = 1; %a counter (handle index) to know where to place the next value

%in the above arrays

while keep\_going

%see if the stop button has been pressed (to stop the loop)

% stop\_btn\_data = get(handles.stop\_btn,'UserData'); %grab the stop\_btn user data

%keep\_going = ~stop\_btn\_data.stop; %when stop button is pressed, its stop\_btn\_data.stop value is changed from 0 to 1, so ~1 will be 0, making keep\_going false, thereby

%forcing the while loop to stop

%lancer le modèle

sim('mesurePressionDanfossAKS.mdl');

temps = toc;% enregistrement du temps écoulé depuis de début de l'acquisition

toto=[mesures(1,1),mesures(1,2),mesures(1,3),mesures(1,4),mesures(1,5),mesures(1,6),temps];

%plot\_handles(handle\_i,1)=loop\_i;

%plot\_handles(handle\_i,2)=toto(1);

%recherche valeur maximale

for i=1:1:2;

if toto(i)>valeurMaxiPression

valeurMaxiPression=toto(i)+5;

end

end

for i=3:1:6;

if toto(i)>valeurMaxiTemperature

valeurMaxiTemperature=toto(i)+5;

end

end

% plot data

% limite des graphiques

xlim(handles.temperature,[0,loop\_i]);

ylim(handles.temperature,[0,valeurMaxiTemperature]);

xlim(handles.pression,[0,loop\_i]);

ylim(handles.pression,[0,valeurMaxiPression]);

% end

%plot mesures

plot(handles.temperature,loop\_i,toto(3),'\*b',loop\_i,toto(4),'+red',loop\_i,toto(5),'\*black',loop\_i,toto(6),'+b');

plot(handles.pression,loop\_i,toto(1),'\*b',loop\_i,toto(2),'+r');

set(handles.textTime,'String',num2str(temps));

handle\_i=handle\_i+1;

%fin graphique

fomat\_donnees='';%initialisation de la mise en forme des mesures dans le fichier

%Première boucle: initialisation de l'acquisition des mesures

if loop\_i==1

%plot the legend

%Write the header to the datalogging file

%General Header notes

fprintf(fid,['Acquisition des températures et pression'...

'\n\n']);

%Column Numbers

for i=1:1:length(toto)

fprintf(fid,'%d,',i);

if i==length(toto)

fprintf(fid,'\n');

end

end

%Print Column Names

fprintf(fid,'BP,HP, T°1 , T°2 , T°3 , T°4, temps');

fprintf(fid,'\n');

%one-time format string creation

format\_string = ''; %initialize string

for i = 1:1:length(toto);

format\_string = strcat(format\_string,'%3.2f'); %append the proper # of format strings

if i < length(toto)

format\_string = strcat(format\_string,','); %add in the commas for all but the last one

else %i = length(data\_to\_log)

format\_string = strcat(format\_string,'\n'); %add in the new line command

end

end

end %end of if loop\_i==1

%Actually log the data into the file (once per flight controller loop!)

fprintf(fid,format\_string,toto);

loop\_i = loop\_i + 1; %incrementation du compteur de boucle

%attente entre deux mesures

pause(60\*1);

%if toc>50

% keep\_going=false;

%end

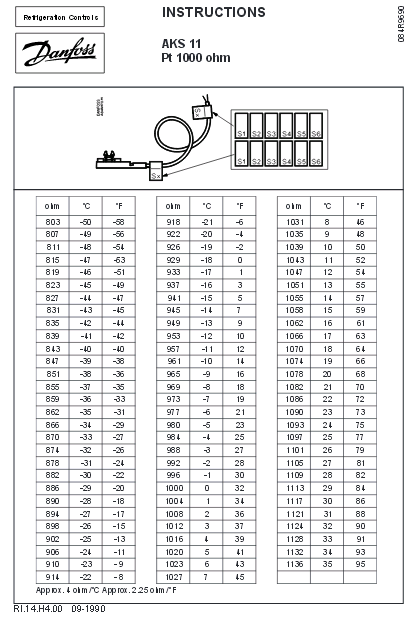
end

fclose(fid); %close the datalogging file

delete(instrfind({'Port'},{'COM11'}));% fermeture de la liaison COM11

# Les capteurs

## Capteur de température : Danfoss AKS-11



## Capteur de pression : Danfoss AKS 32R

