

Rapport de Laboratoire - Control Theory

Nicolas Jurquet (21211) - Xavier Allaud(21151)

March 2024

1 Introduction

Le but de ce rapport est d'expliquer le processus de conception d'un régulateur PID avec une boucle de rétroaction en utilisant un kit TCLab. Pour se faire, il faudra identifier la dynamique d'entrée/sortie du système afin de construire un modèle de celui-ci. Sur base de ce modèle, il sera possible d'implémenter un PID discret qu'il faudra ensuite optimiser puis tester en conditions réelles.

Table des matières

1	Introduction	1
2	Identification de la Dynamique	2
2.1	Processus P(s)	2
2.1.1	Approximation graphique du modèle pour MV	3
2.1.2	Modèle de Borda (FOPDT)	3
2.1.3	Modèle de van der Grinten (SOPDT)	4
2.1.4	Modèle de Strejc	4
2.1.5	Comparaison des modèles d'approximation	5
2.2	Perturbation D(s)	6
2.2.1	Comparaison des modèles avec paramètres optimaux	7
3	Régulateur PID	9
3.1	Introduction	9
3.2	Optimisation par la méthode IMC	9
3.3	Réponse indicelle du régulateur PID	9
3.3.1	Influence de K_C	11
3.3.2	Influence de T_D et T_I	11
3.3.3	Influence de α	11
3.4	FeedForward	11
3.4.1	Délai	12
3.4.2	Gain et Lead-Lag	12
4	Données Expérimentales	14
4.1	Scénario 1	15
4.2	Scénario 2	16
4.3	Scénario 3	17
A	Méthode graphique pour l'obtention des paramètres d'approximation	18

2 Identification de la Dynamique

Pour ce qui va être présenté par la suite ait du sens, il est nécessaire de savoir dans quelles conditions nous travaillons.

Nos réponses sont basées sur un point de fonctionnement de $MV_0 = 50\%$, $DV_0 = 50\%$ et une sortie $PV_0 = 49.3^\circ\text{C}$. C'est à dire que la température obtenue sur PV lorsqu'une puissance de chauffage de 50% est appliquée sur les 2 chauffages est de 49.3°C en régime établi.

2.1 Processus $P(s)$

Appliquons un step $X(s) = \frac{1}{s}$ autour du point de fonctionnement (de 30% à 70%) lorsque le système se situe en régime établi pour analyser la "step response" $Y(s) = \frac{P(s)}{s}$.

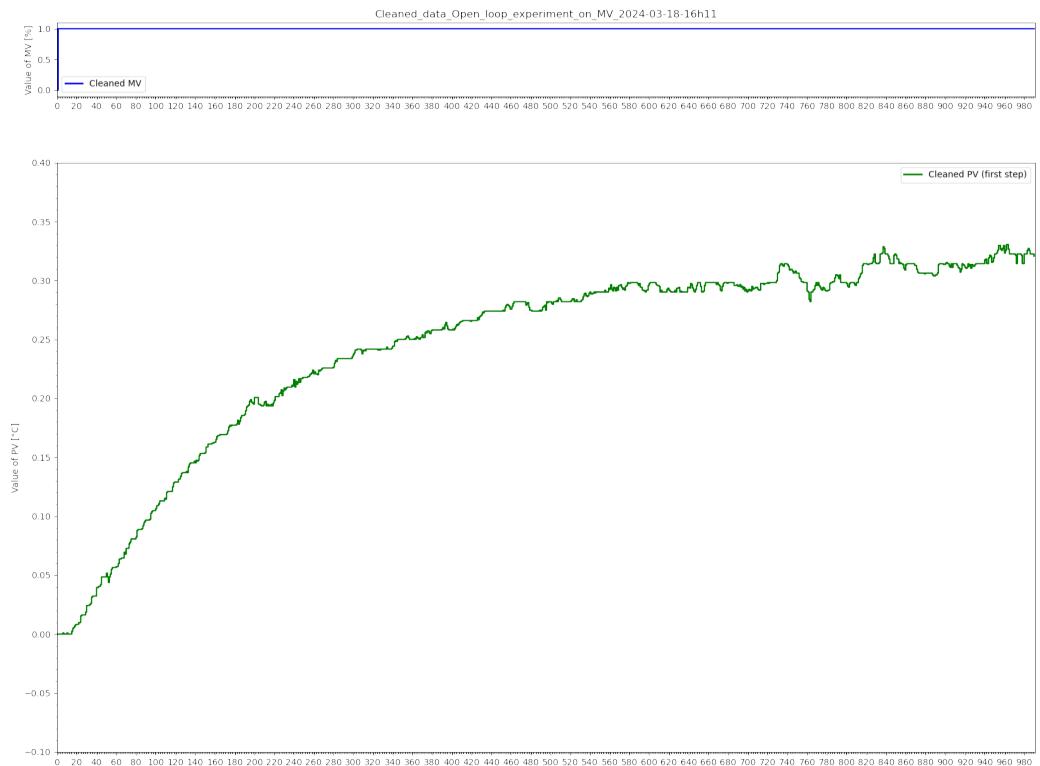


FIGURE 1 – Réponse à un step sur MV

- Le graphique (Figure 1) obtenu permet (presque) d'affirmer que le processus est un système du 1^e ordre avec un délai. L'allure de la "step response" est donc $y(t) = K_p(1 - e^{-t/T})u(t)$.
- Le but est maintenant d'obtenir les paramètres de la fonction de transfert du processus $P(s)$ ce qui permettra de prédire $Y(s)$ pour n'importe quel $X(s)$ puisque $Y(s) = P(s) \cdot X(s)$ ou $PV = P(s) \cdot MV$. Pour se faire, on utilise une fonction de minimisation d'erreur :

```
minimize(FOPDT_cost,
    p0,args=(MVm,PVm,Ts,
    (fig,ax1,l1,l2)),
    method='Powell',
    bounds=bdns,
    options={'maxiter': maxIter})
```

Où $p0$ contient les points de départ d'optimisation pour les paramètres K_p , T et θ . La fonction de minimisation va faire varier ces paramètres afin que la fonction de coût, $FOPDT_cost$, renvoie la valeur la plus basse possible.

$FOPDT_cost$:

```

for i in range(0,len(MV)):
    t.append(i*Ts) #calcule le temps correspondant au point actuel
    ↪ de la donnée en se basant sur la période de sample
    MVTemp.append(MV[i]) #itère sur la longueur du vecteur d'entrée

Delay_RT(MVTemp,theta,Ts,MVDelay) #application d'un délai à MV
FO_RT(MVDelay,Kp,T,Ts,PVSim) #On applique un délai au premier ordre via le
↪ délai appliqué à MV puis on calcule un point du vecteur de sortie à
↪ chaque itération
objective = objective + (PV[i] - PVSim[i])**2 #on ajoute le carré de
↪ l'erreur actuel à la somme des carré des erreurs précédentes

```

Le fichier `Identification.ipynb` permet de trouver les valeurs estimées de K_P , T_{1p} , T_{2p} et θ_p qui vont par après nous servir à modéliser le Processus et la Perturbation de façon optimale, ainsi que les différentes méthodes d'approximations des modèles du 1^e et 2^e ordre.

Pour notre réponse du Processus, nous obtenons les valeurs suivantes pour un modèle du 2^e ordre :

$$\begin{aligned} K_P &= 0.308 \\ T_{1p} &= 183.819 \text{ s} \\ T_{2p} &= 3.292 \cdot 10^{-12} \text{ s} \\ \theta_p &= 20.015 \text{ s} \end{aligned}$$

Et les valeurs suivantes pour un modèle du 1^{er} ordre :

$$\begin{aligned} K_P &= 0.314 \\ T_p &= 206.264 \text{ s} \\ \theta_p &= 12.999 \text{ s} \end{aligned}$$

On remarque directement que T_{2p} est négligeable par rapport à T_{1p} , et permet donc de confirmer que le processus agit comme un système du 1^{er} ordre avec délai :

$$\hat{P}(s) = \frac{K_P e^{-\theta_p s}}{(T_{1p}s + 1)(T_{2p}s + 1)} \approx \frac{K_P e^{-\theta_p s}}{T_{1p}s + 1} \quad (1)$$

2.1.1 Approximation graphique du modèle pour MV

Il est également possible de déterminer la dynamique du Processus graphiquement sur base de la Figure 1. Nous obtenons les paramètres K_P , T_u , T_g , t_1 , t_2 et a dont la détermination est expliquée en Annexe A :

$$\begin{aligned} K_P &= 0.305 \\ T_u &= 17 \text{ s} \\ T_g &= 211 \text{ s} \\ t_1 &= 95 \text{ s} \\ t_2 &= 133 \text{ s} \\ a &= 0.1 \end{aligned}$$

Ces paramètres serviront à utiliser les méthodes classiques d'approximation du modèle, à savoir le modèle de **Broida**, **Van Der Grinten** et **Strejc**.

2.1.2 Modèle de Broida (FOPDT)

Ce modèle consiste à approximer le Processus par un système du 1^{er} ordre avec délai de la forme :

$$P_B(s) = \frac{K_P e^{-\theta s}}{Ts + 1}$$

Le 1^{er} modèle de Broida est obtenu en définissant la constante de temps T et le délai θ comme suit :

$$T = T_g = 211 \text{ s} \quad \text{et} \quad \theta = T_u = 17 \text{ s}$$

Le 2^e modèle de Broida est obtenu en définissant la constante de temps T et le délai θ comme suit :

$$T = 5.5(t_2 - t_1) = 209 \text{ s} \quad \text{et} \quad \theta = 2.8t_1 - 1.8t_2 = 26.60 \text{ s}$$

2.1.3 Modèle de van der Grinten (SOPDT)

Ce modèle consiste à approximer le Processus par un système du 2^e ordre avec délai de la forme :

$$P_{vdG}(s) = \frac{K_P e^{-\theta s}}{(T_1 s + 1)(T_2 s + 1)}$$

T_1 , T_2 et θ sont obtenus comme suit :

$$\begin{aligned} T_1 &= T_g \frac{3ae - 1}{1 + ae} = -30.61 \text{ s} \\ T_2 &= T_g \frac{1 - ae}{1 + ae} = 120.81 \text{ s} \\ \theta &= T_u - \frac{T_1 T_2}{T_1 + 3T_2} = 28.15 \text{ s} \end{aligned}$$

Nous constatons que la 1^e constante de temps T_1 est négative, ce qui est physiquement impossible ! Il n'est cependant pas étonnant d'obtenir ce résultat étant donné que le Processus est un système du 1^{er} ordre avec délai. Un modèle du 2^e ordre tel que van der Grinten n'est donc pas adapté pour approximer notre Processus et ne sera donc pas représenté par après.

2.1.4 Modèle de Strejc

Ce modèle consiste à approximer le Processus par un système du n^e ordre avec des pôles et constantes de temps identiques de la forme :

$$P_S(s) = \frac{K_P e^{-\theta s}}{(Ts + 1)^n}$$

L'ordre n est obtenu avec la Table 1 :

Order n	$T_{u_{th}}/T_g$	T_g/T
	a_n	b_n
1	0.00	1.00
2	0.10	2.72
3	0.22	3.69
4	0.32	4.46
5	0.41	5.12
6	0.49	5.70
7	0.57	6.23

TABLE 1 – Ordre n du modèle de Strejc en fonction de a_n et b_n

Nous avons que $T_u/T_g = 0.08$ et donc ce situe à $a_1 \leq 0.08 < a_2$, ce qui nous donne un ordre $n = 1$. a_n vaut alors 0.00 et b_n vaut 1.00. La constante de temps T et le délai θ vont être au final trouvés directement avec les valeurs de T_g et T_u comme le modèle de Broida :

$$\begin{aligned} T &= \frac{T_g}{b_n} = T_g = 211 \text{ s} \\ \theta &= T_u - a_n T_g = T_u = 17 \text{ s} \end{aligned}$$

2.1.5 Comparaison des modèles d'approximation

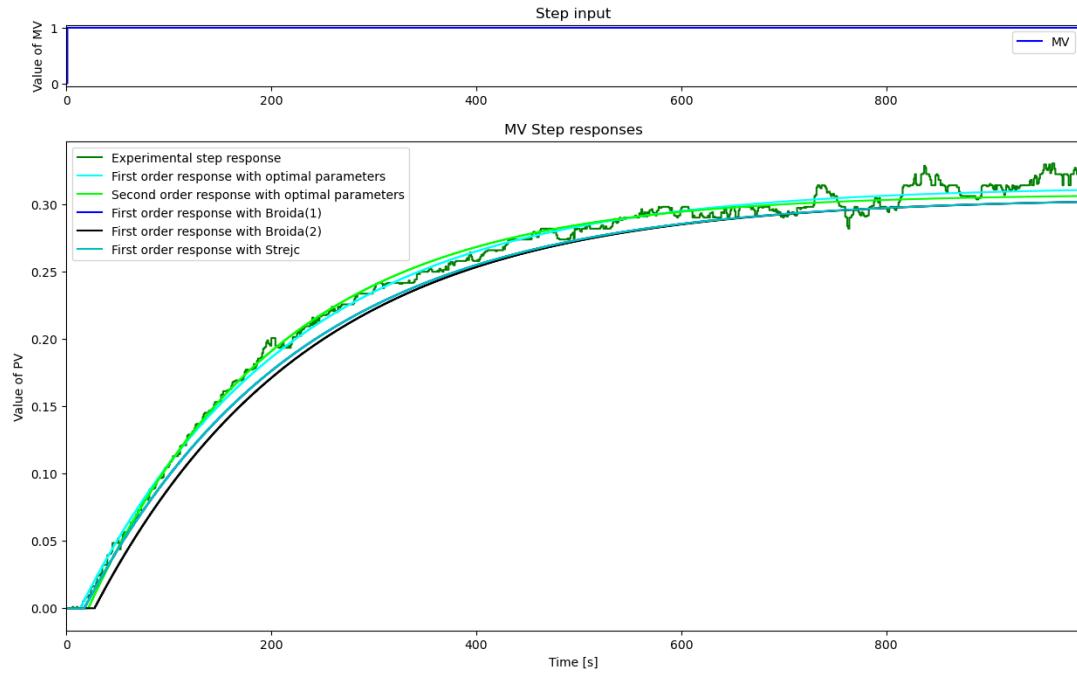


FIGURE 2 – Approximations de la réponse temporelle d'un step sur MV

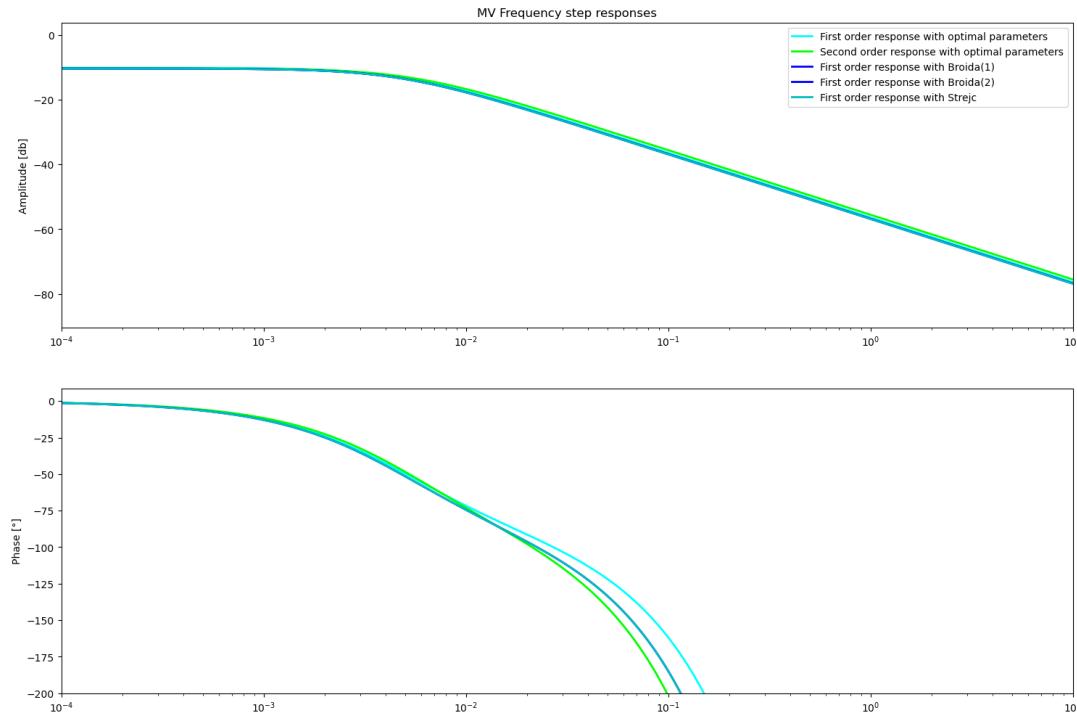


FIGURE 3 – Approximations de la réponse fréquentielle d'un step sur MV

La Figure 2 montre les différentes réponses d'un step sur MV pour chacun des modèles décrits au-dessus (à l'exception du modèle de van der Grinten). Elles représentent toutes bien une réponse du 1^e ordre avec délai, et restent assez proches les unes des autres. Les modèles commencent tous avec un délai similaire et terminent en régime établi à environ le même gain statique K_P . Cependant, on distingue tout de même les modèles avec paramètres optimaux des modèles graphiques.

Les paramètres optimaux ont été trouvés par un algorithme de minimisation de l'erreur, tandis que les paramètres graphiques ont été trouvés par une méthode graphique dont la précision est à prendre avec recul.

La Figure 3 nous permet ensuite de comparer les réponses fréquentielles des différents modèles. La première observation est que tous les modèles sont pratiquement superposés en terme de gain. En basses fréquences, ils atteignent bien un gain de $K_P = -10 \text{ dB} = 0.3$ et en hautes fréquences, ils subissent tous une pente de -20 dB/décade . On peut également confirmer que la réponse du 2^e ordre avec paramètres optimaux est bien en fait un 1^{er} ordre en raison d'une constante de temps négligeable.

En ce qui concerne la phase, on obtient bien une allure à laquelle on s'attendait pour un système du 1^{er} ordre avec délai, à savoir une phase qui tends vers l'infini (en négatif) lorsque la fréquence tends vers l'infini. Nous constatons que la courbe bleue (premier ordre optimal) et la courbe verte (second ordre optimal) dévient plus la fréquence augmente. Cela peut être expliqué par la différence entre les deux délais θ_p qui sont de 13 et 20 secondes respectivement. En effet, pour atteindre une même phase représentée par $\theta_s = j\theta\omega$, il faudra une fréquence plus élevée pour le modèle du 1^{er} ordre que pour le modèle du 2^e ordre.

2.2 Perturbation D(s)

Nous appliquons maintenant un step sur DV pour observer la réponse du système à une perturbation.



FIGURE 4 – Réponse à un step sur DV

Étant donné l'allure du graphe (Figure 4), et en particulier le point d'inflexion aux alentours de 80 secondes, nous pouvons estimer que la perturbation se comporte comme un système du 2^e ordre avec délai de la forme :

$$\hat{D}(s) = \frac{K_D e^{-\theta_d s}}{(T_{1d}s + 1)(T_{2d}s + 1)} \quad (2)$$

En effet, nous obtenons grâce au fichier `Identification.ipynb`, les valeurs optimales suivantes pour un

modèle du 2^e ordre :

$$\begin{aligned} K_D &= 0.295 \\ T_{1d} &= 182.255 \text{ s} \\ T_{2d} &= 13.184 \text{ s} \\ \theta_d &= 28.999 \text{ s} \end{aligned}$$

Et les valeurs suivantes pour un modèle du 1^{er} ordre :

$$\begin{aligned} K_D &= 0.296 \\ T_d &= 184.880 \text{ s} \\ \theta_d &= 40.136 \text{ s} \end{aligned}$$

Les valeurs obtenues pour T_{1d} et T_{2d} nous permettent de confirmer que la perturbation peut se comporter comme un système du 2^e ordre.

2.2.1 Comparaison des modèles avec paramètres optimaux



FIGURE 5 – Approximations de la réponse temporelle d'un step sur DV

Il est clair que la Figure 5 montre que les modèles du 1^{er} et 2^e ordre avec paramètres optimaux sont très proches l'un de l'autre. La seule différence réside en la décomposition du délai du 1^{er} ordre en un délai et une constante de temps T_{2d} pour le 2^e ordre. L'ajout de cette constante de temps permet de mieux modéliser la réponse expérimentale du système à une perturbation.



FIGURE 6 – Approximations de la réponse fréquentielle d'un step sur DV

Pouvant modéliser la perturbation comme un système du 1^{er} ou 2^e ordre avec délai, la réponse fréquentielle (Figure 6) est différente pour les deux modèles. À hautes fréquences, le modèle du premier ordre possède une pente de -20 dB/décade tandis que le modèle du second ordre possède une pente de -40 dB/décade . Le gain à basses fréquences vaut bien également $K_P = -10 \text{ dB} = 0.3$.

La phase, comme pour la partie Processus, tends vers l'infini (en négatif) lorsque la fréquence tends vers l'infini et dévie entre les deux modèles à hautes fréquences. Cela peut être expliqué par la différence entre les deux délais θ_p qui sont cette fois de 40 secondes pour le premier ordre et 29 secondes pour le second ordre. Le délai du second ordre étant cette fois plus petit, nous avons le comportement opposé : pour atteindre une même phase représentée par $\theta s = j\theta\omega$, il faudra une fréquence plus élevée pour le modèle du 2^e ordre que pour le modèle du 1^{er} ordre.

3 Régulateur PID

3.1 Introduction

Afin de réguler les variations en sortie PV , on utilise un régulateur PID, qui reprend la valeur de PV pour la soustraire à la consigne SP donnant donc l'erreur $E = SP - PV$ à corriger sur MV .

Pour rappel, un régulateur PID est composé de trois termes :

- Le terme proportionnel P qui est proportionnel à l'erreur E et vise une erreur statique nulle.
- Le terme intégral I qui est proportionnel à la somme des erreurs passées et donc accumule l'erreur.
- Le terme dérivé D qui est proportionnel à la dérivée de l'erreur et vise à corriger anticipativement l'erreur future.

La sortie du régulateur est alors donnée par :

$$MV = K_C \left(1 + \frac{1}{T_I s} + \frac{T_D s}{\alpha T_D s + 1} \right) E \quad (3)$$

Dans le cadre du laboratoire, le régulateur utilise également le **Reset de l'Action Intégrale** et la **Saturation de l'Action Intégrale** venant adapter l'action intégrale en fonction de, respectivement, la valeur de MV en mode manuel, et la saturation de MV atteignant les limites MV_{MAX} / MV_{MIN} .

$$MV_I = MV_{Man} - MV_P - MV_D - MV_{FF}$$

et

$$MV_I = MV_{MAX} - MV_P - MV_D - MV_{FF}$$

3.2 Optimisation par la méthode IMC

Il est important de choisir les paramètres K_C , T_I et T_D de façon à implémenter le bon régulateur pour notre processus. Une façon d'obtenir ces paramètres optimaux est de réaliser un step sur MV et d'observer la dynamique du Processus. Le modèle trouvé va nous permettre de calculer ces valeurs via des tables.

On utilisera la ligne I du tableau présent dans le cours, correspondant à un modèle du second ordre avec délai ($\tau_3 = 0$).

$$\begin{aligned} K_C &= \frac{1}{K_P} \frac{T_{1p} + T_{2p}}{T_{CLP} + \theta} \\ T_I &= T_{1p} + T_{2p} \\ T_D &= \frac{T_{1p} T_{2p}}{T_{1p} + T_{2p}} \end{aligned}$$

Il est bon de noter que nous aurions pu utiliser la ligne G du tableau (premier ordre avec délai) totalement équivalente étant donné que notre Processus est du premier ordre ($T_{2p} \approx 0$). La constante T_D et donc l'action Dérivée valant 0, le régulateur devient en fait un régulateur PI.

$$\begin{aligned} K_C &= \frac{1}{K_P} \frac{T_{1p}}{T_{CLP} + \theta} \\ T_I &= T_{1p} \\ T_D &= 0 \end{aligned}$$

La constante de temps en boucle fermée T_{CLP} est un certain ratio de la première constante de temps du processus T_{1p} définie par $T_{CLP} = \gamma T_{1p}$. L'influence de γ sera discutée en simulation de boucle fermée par après.

3.3 Réponse indicielle du régulateur PID

Nous allons maintenant analyser la réponse du régulateur lorsqu'on applique une erreur E constante à son entrée. La figure 7 représente cette réponse. Premièrement, nous voyons tout au long du graphique que MV est la somme de ces actions $MV = MV_P + MV_I + MV_D + MV_{FF}$. A l'instant du step sur

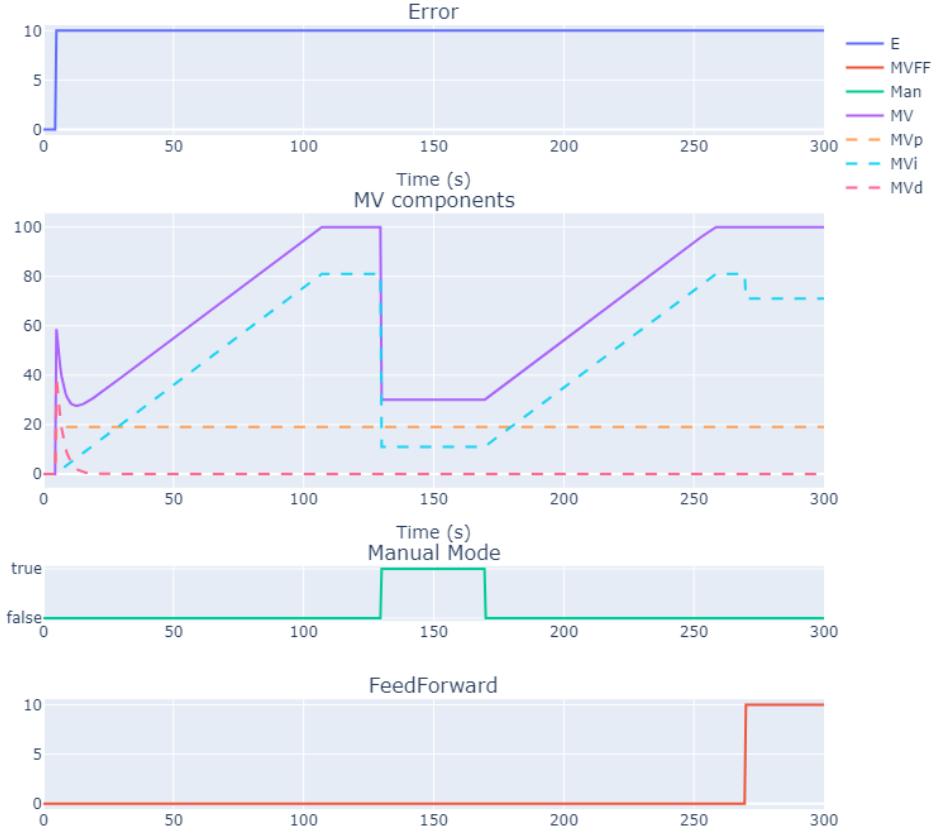


FIGURE 7 – Réponse indicielle du PID à un step sur E

l’erreur, la composante proportionnelle MV_P augmente instantanément puisqu’elle est proportionnelle à l’erreur, la composante intégrale MV_I est encore nulle puisqu’elle n’a pas encore accumulé d’erreurs, et la composante dérivée MV_D est limitée à $1/\alpha$ puis diminue suivant sa constante de temps T_D .

Ensuite, lorsque nous sommes en mode automatique, l’action intégrale MV_I augmente linéairement puisque l’erreur est constante. Cela va continuer jusqu’à éventuellement atteindre les limites imposées sur la sortie MV , ici, une puissance de chauffe $MV_{MAX} = 100\%$ et $MV_{MIN} = 0\%$. Il est alors nécessaire de réaliser un Reset de l’Action Intégrale, c’est-à-dire, venir adapter la valeur de MV_I pour garder la sortie MV dans les limites.

$$MV_I = MV_{MAX} - MV_P - MV_D - MV_{FF}$$

Lorsque l’on passe en mode manuel (boucle ouverte), la valeur de MV est donnée par l’opérateur et donc ici, fixé à 30%. En effet, on voit sur le graphe que MV chute à 30% et que l’action intégrale MV_I est alors adaptée pour garder la sortie MV à cette valeur.

Enfin, une perturbation a été ajoutée à la fin de la simulation pour montrer l’effet de l’action Feed-Forward. À saturation, le Reset de l’Action Intégrale prends également en compte cette perturbation pour garder la sortie MV dans les limites et donc, on le voit sur le graphe lorsque MV_I est réduit de 10%. Si MV ne sature pas, cette perturbation sera directement ajoutée à la sortie MV permettant ainsi de minimiser l’impact sur PV .

3.3.1 Influence de K_C

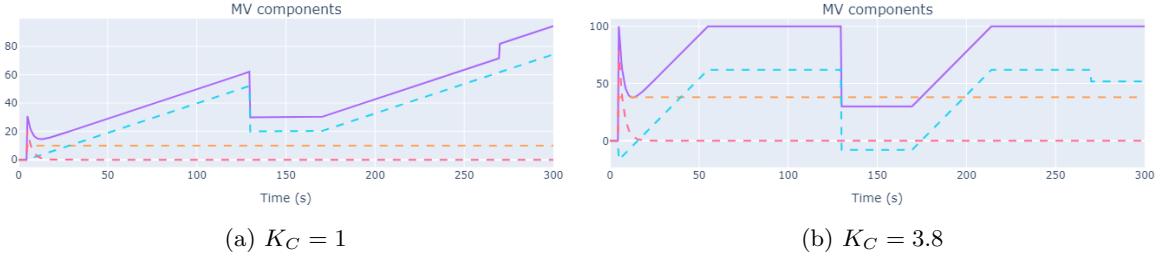


FIGURE 8 – Influence de K_C sur la réponse indicielle du PID

3.3.2 Influence de T_D et T_I

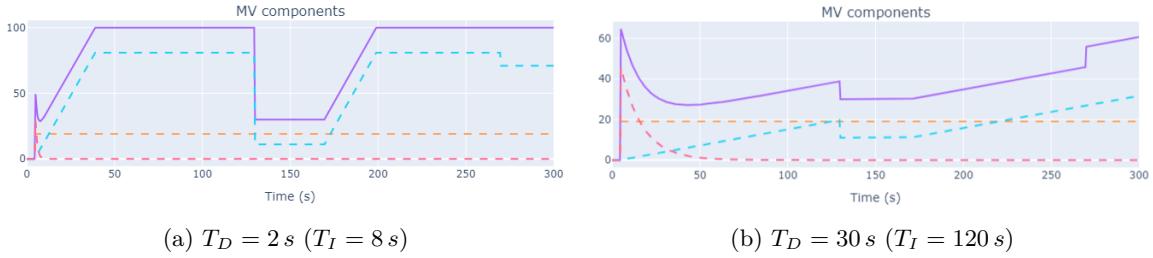


FIGURE 9 – Influence de T_D et T_I sur la réponse indicielle du PID

3.3.3 Influence de α

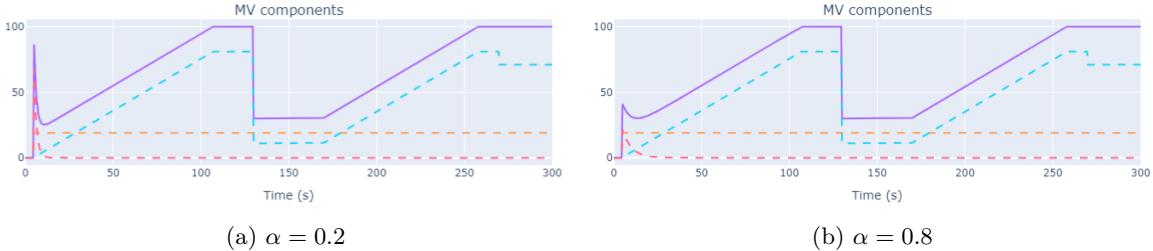


FIGURE 10 – Influence de α sur la réponse indicielle du PID

3.4 FeedForward

La fonction de FeedForward est conçue pour anticiper et compenser l'impact des perturbations (DV) sur la variable du processus (PV), avant que ces dernières n'affectent le système.

Le fonctionnement du FeedForward peut être décrit de manière mathématique comme suit :

- 1 Premièrement, la valeur manipulée en sortie du PID, MV_{FB} , est ajustée par une valeur MV_{FF} , calculée pour compenser directement la perturbation :

$$MV_{FF} = K_{FF} \cdot \frac{(T_{1P}s + 1)(T_{2P}s + 1)}{(T_{1D}s + 1)(T_{2D}s + 1)} \cdot e^{-\theta_{FF}} DV = \frac{\hat{D}(s)}{\hat{P}(s)} DV \quad (4)$$

où :

$$K_{FF} = \frac{K_D}{K_P}, \quad (5)$$

$$\theta_{FF} = |\theta_D - \theta_P| \quad (6)$$

- 2 Ensuite, après la sortie du nœud $MV = MV_{FB} - MV_{FF}$, on obtient :

$$P(s) \cdot MV \approx P(s) \cdot MV_{FB} - \hat{D}(s) \quad (7)$$

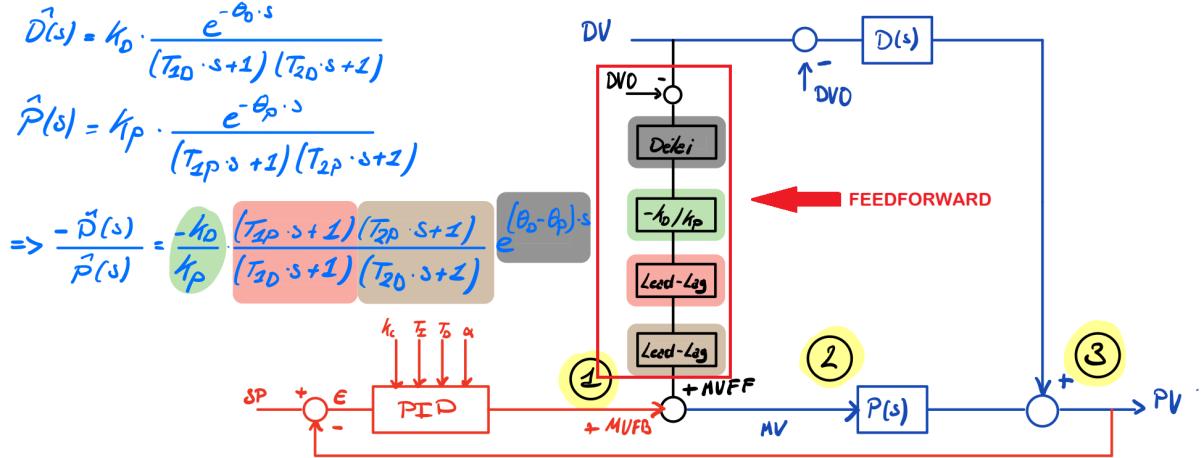


FIGURE 11 – Schema du régulateur PID avec fonction de FeedForward

- 3 Pour enfin arriver au nœud final où l'on additionne la dynamique de la perturbation $D(s)$ à celle du processus :

$$P(s) \cdot MV_{FB} - \hat{D}(s) + D(s) \approx P(s) \cdot MV_{FB} = PV \quad (8)$$

3.4.1 Délai

La 1^{ère} étape dans la réalisation de la fonction du FeedForward est de récupérer la perturbation DV , de la ramener au point de fonctionnement et de lui appliquer un délai θ_{FF} (6). L'application de ce délai se fait à l'aide de la fonction `Delay_RT` :

```
Delay_RT(self.DV - self.DV0*np.ones_like(self.DV), # On recentre DV sur le point de
         fonctionnement
         max(self.theta_ODV_SOPDT-self.theta_OMP_SOPDT, 0), # Calcul du délai
         self.Ts,
         self.MVFF_Delay)
```

3.4.2 Gain et Lead-Lag

Après avoir appliqué le délai à la perturbation et l'avoir recentrée pour la ramener au point de fonctionnement, l'étape suivante consiste à appliquer le gain (5) et le premier Lead-Lag.

Ajouter un signe négatif au gain puis additionner le MV_{FF} au MV_{FB} ou laisser le gain positif et soustraire MV_{FF} au MV_{FB} est sans conséquences sur le système.

Le premier Lead-Lag dont $T_{Lead} = T_{1P}$ et $T_{Lag} = T_{1D}$ couplé au gain et au délai permet d'obtenir la fonction de transfert :

$$K_{FF} \frac{T_{1P}s + 1}{T_{1D}s + 1} e^{\theta_{FF}}$$

Pour se faire, on utilise la fonction `LL_RT` sur le signal DV retardé et centré (MV_{Delay}) via :

```
LL_RT(self.MVFF_Delay,
      -self.Kp_ODV_SOPDT/self.Kp_OMP_SOPDT, # gain
      self.T1_OMP_SOPDT, # TLead
      self.T1_ODV_SOPDT, # TLag
      self.Ts,
      self.MVFF_LL1)
```

On applique ensuite un second Lead-Lag afin d'obtenir la fonction de transfert complète (4) dont la constant $T_{Lead} = T_{2P}$ et $T_{Lag} = T_{2D}$.

Dans le cas où le FeedForward est désactivé, on applique un gain nul au second Lead-Lag donc $MV_{FF} = 0$:

```
if self.FF == True:
    LL_RT(self.MVFF_LL1,
           1, # Gain unitaire quand le FeedForward est activé
           self.T2_0MV_SOPDT, #TLead
           self.T2_0DV_SOPDT, #TLag
           self.Ts,
           self.MVFF
    )
else:
    LL_RT(self.MVFF_LL1,
           0, # Gain nul quand le FeedForward est désactivé
           self.T2_0MV_SOPDT,
           self.T2_0DV_SOPDT,
           self.Ts,
           self.MVFF
    )
```

4 Données Expérimentales

Nous allons maintenant tester notre régulateur PID sur la plateforme afin de comparer son comportement en simulation à son comportement réel. Nous avons donc mis en place 4 scénarios qui permettront de comparer tous les cas de figure :

1. Le 1^{er} scénario permet de tester le fonctionnement du régulateur en boucle ouverte (mode manuel) sans FeedForward en observant sa réponse à un step de 10% sur *DV*.

```
SPPath = {0: PVO}
ManPath = {0: True, TSim: True} # Mode manuel actif toute la simulation
MVManPath = {0: MVO, TSim: MVO} # MV manuel reste constant
DVPath = {0: DVO, 1300: DVO + 10} # Step de 10% après 1300 secondes
FF = False # Pas de FeedForward
ManFF = False # Pas de prise en compte du FeedForward
```

2. Le 2^{ème} scénario permet de tester le fonctionnement du régulateur en boucle ouverte et avec FeedForward en observant sa réponse à un step de 10% sur *DV*.

```
SPPath = {0: PVO}
ManPath = {0: True, TSim: True} #Mode manuel actif toute la simulation
MVManPath = {0: MVO, TSim: MVO} # MV manuel reste constant
DVPath = {0: DVO, 1300: DVO + 10} # Step de 10% après 1300 secondes
FF = True # FeedForward activé
ManFF = True # Prise en compte du FeedForward
```

3. Le 3^{ème} scénario permet de tester le fonctionnement du régulateur en boucle fermée (mode automatique) sans FeedForward en observant sa réponse à un step de -10% sur *SP* puis un step de 10% sur *DV*.

```
SPPath = {0: PVO + 5, 1000: PVO - 5} # Step de -10% après 1000 secondes
ManPath = {0: True, 500: False, TSim: False}
MVManPath = {0: MVO+15, TSim: MVO+15}
DVPath = {0: DVO, 1600: DVO + 10} # Step de 10% après 1600 secondes
FF = False # Pas de FeedForward
ManFF = False
```

4. Le 4^{ème} scénario permet de compléter le scénario précédent en testant le fonctionnement du régulateur en boucle fermée FeedForward en observant sa réponse à un step de -10% sur *SP* puis à un step de 10% sur *DV*.

```
SPPath = {0: PVO + 5, 1000: PVO - 5} #Step de -10% après 1000 secondes
ManPath = {0: True, 500: False, TSim: False} # Mode automatique après 500
    ↵ secondes
MVManPath = {0: MVO+15, TSim: MVO+15}
DVPath = {0: DVO, 1600: DVO + 10} # Step de 10% après 1600 secondes
FF = True # FeedForward activé
ManFF = False
```

4.1 Scénario 1

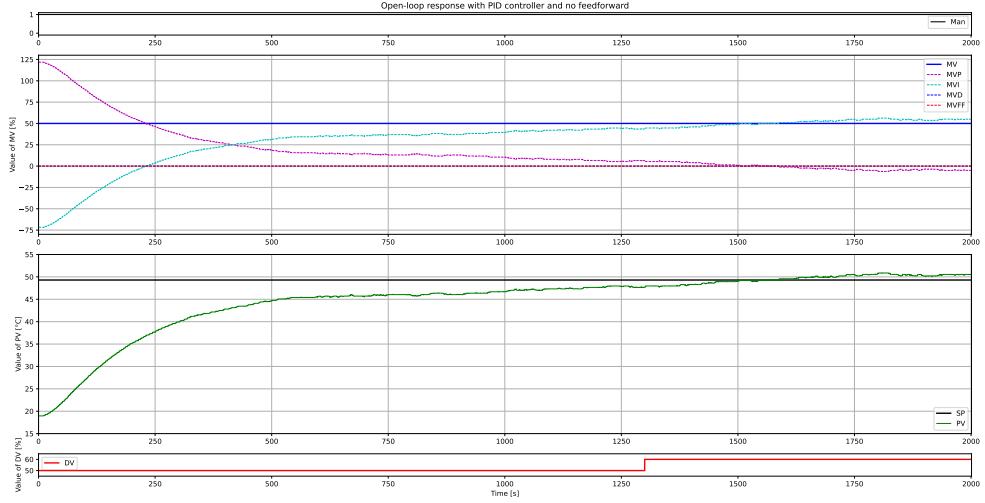


FIGURE 12 – Données expérimentales du 1^{er} scénario

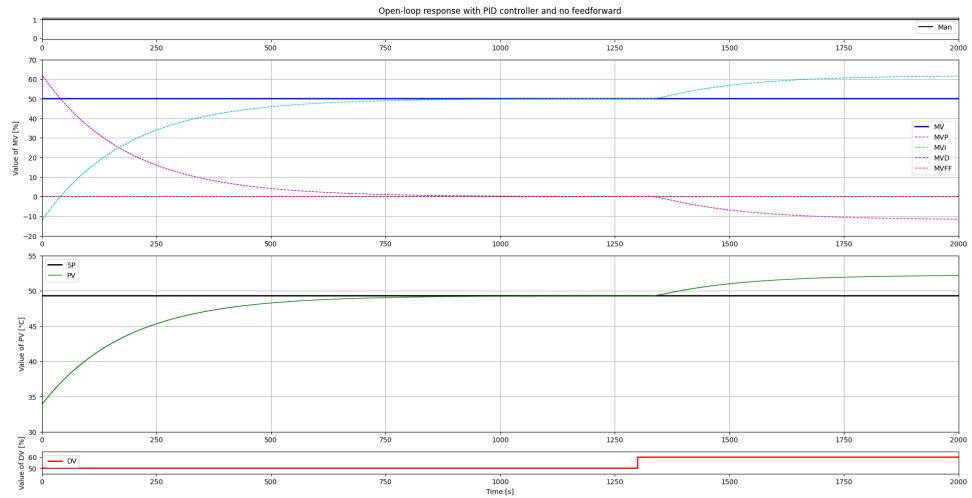


FIGURE 13 – Données de la simulation du 1^{er} scénario

On peut observer que le *PV* n'a pas atteint la valeur de consigne (*SP*) avant le step de 10% sur *DV*. Cela est dû au fait que, pour l'expérience, la valeur du *PV* à $t = 0s$ est de $\sim 18^{\circ}C$ tandis que pour la simulation, il est de $\sim 34^{\circ}C$. Néanmoins, une fois le step de 10% sur *DV* effectué, la valeur du *PV* converge jusqu'à la valeur de consigne et la dépasse à $t = 1500s$ ce qui correspond à un retard par rapport à la simulation $\sim \Delta t = 75s$.

Une autre observation importante concerne la forme de la courbe *PV* qui suggère un comportement typique d'un système du second ordre. Cependant, l'analyse des dynamiques du système avait révélé une seconde constante de temps très faible, $T_2P \sim 10^{-12}$, négligeable, ce qui a conduit à la conclusion que notre système se comportait plutôt comme un système du première ordre en réponse à un changement sur *MV*. Selon nous, une des causes serait la température plus basse de la pièce qui affecterait l'inertie thermique du capteur de température. L'humidité de la pièce pourrait aussi être facteur puisque celle-ci affecte la conductivité thermique de l'air ambiant et donc pourrait aussi impacter l'inertie thermique du capteur.

4.2 Scénario 2

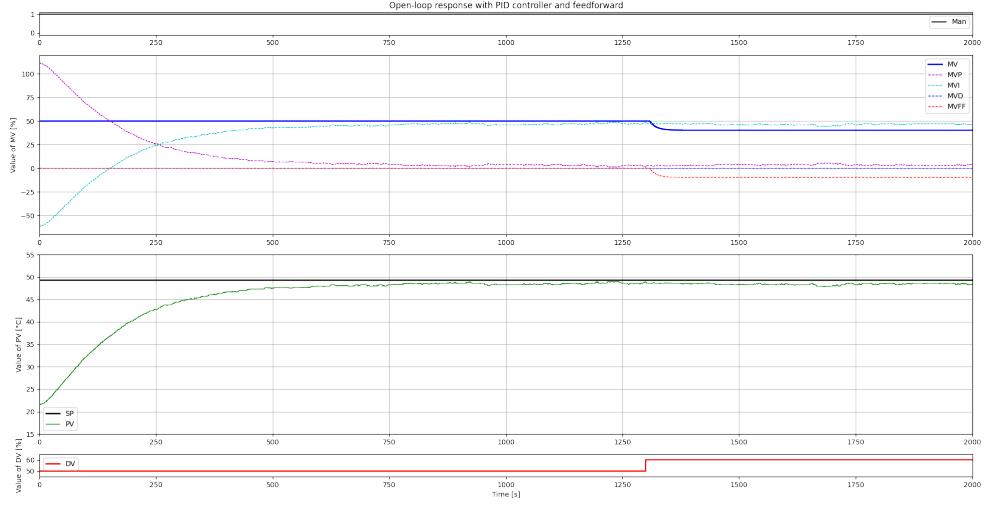


FIGURE 14 – Données expérimentales du 2^{ème} scénario

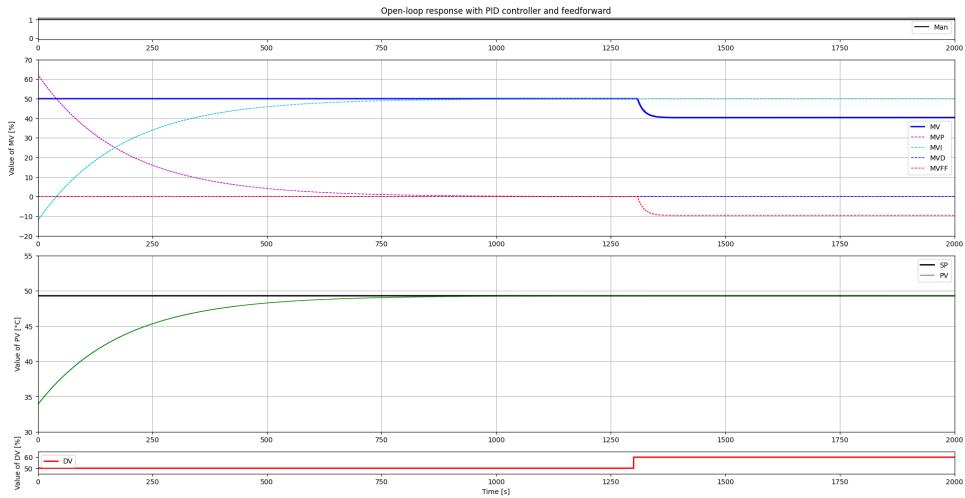


FIGURE 15 – Données de la simulation du 2^{ème} scénario

Comme pour le scénario précédent(12), on peut voir que le système régit plutôt comme un système du second ordre.

Cependant, contrairement à l'exemple précédent, la valeur du *PV* atteint à peu près au même moment la valeur de consigne (*SP*) que pour la simulation. Ceci est dû au fait que la température du capteur n'a pas suffisamment pu redescendre entre les deux scénarios. En effet, la valeur de *PV* à $t = 0s$ est de $\sim 22^{\circ}C$ ce qui correspond à $4^{\circ}C$ de plus que pour le scénario précédent.

À part cela, le système réagit bien à la perturbation grâce au FeedForward qui permet de compenser entièrement le step de 10% sur *DV*.

4.3 Scénario 3

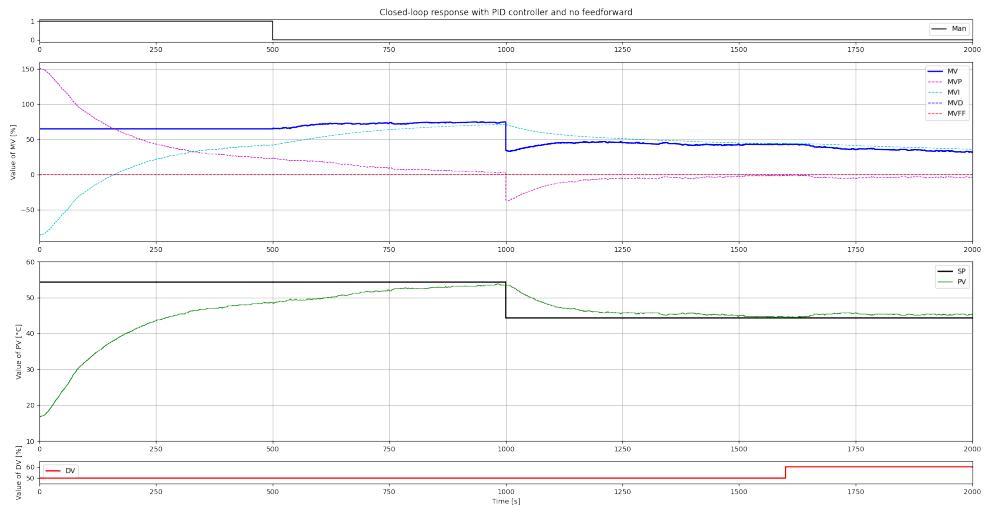


FIGURE 16 – Données expérimentales du 3^{ème} scénario

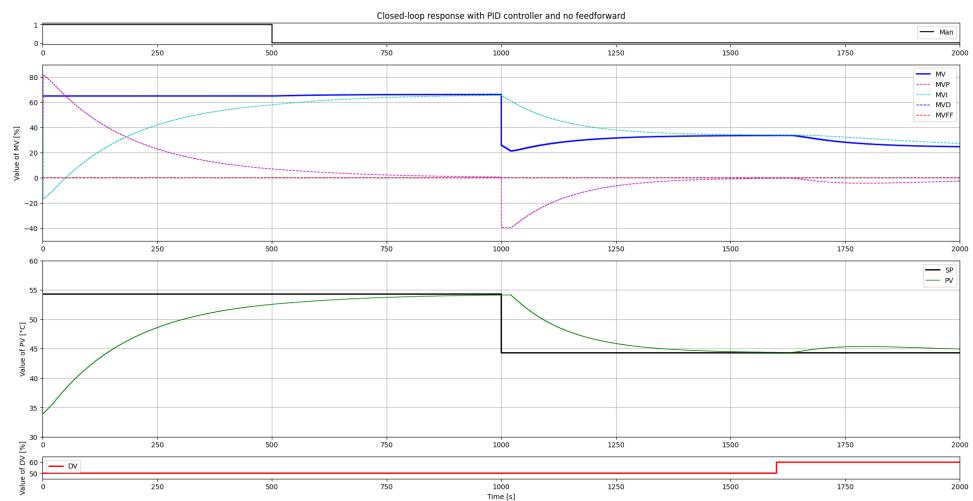


FIGURE 17 – Données de la simulation du 3^{ème} scénario

A Méthode graphique pour l'obtention des paramètres d'approximation

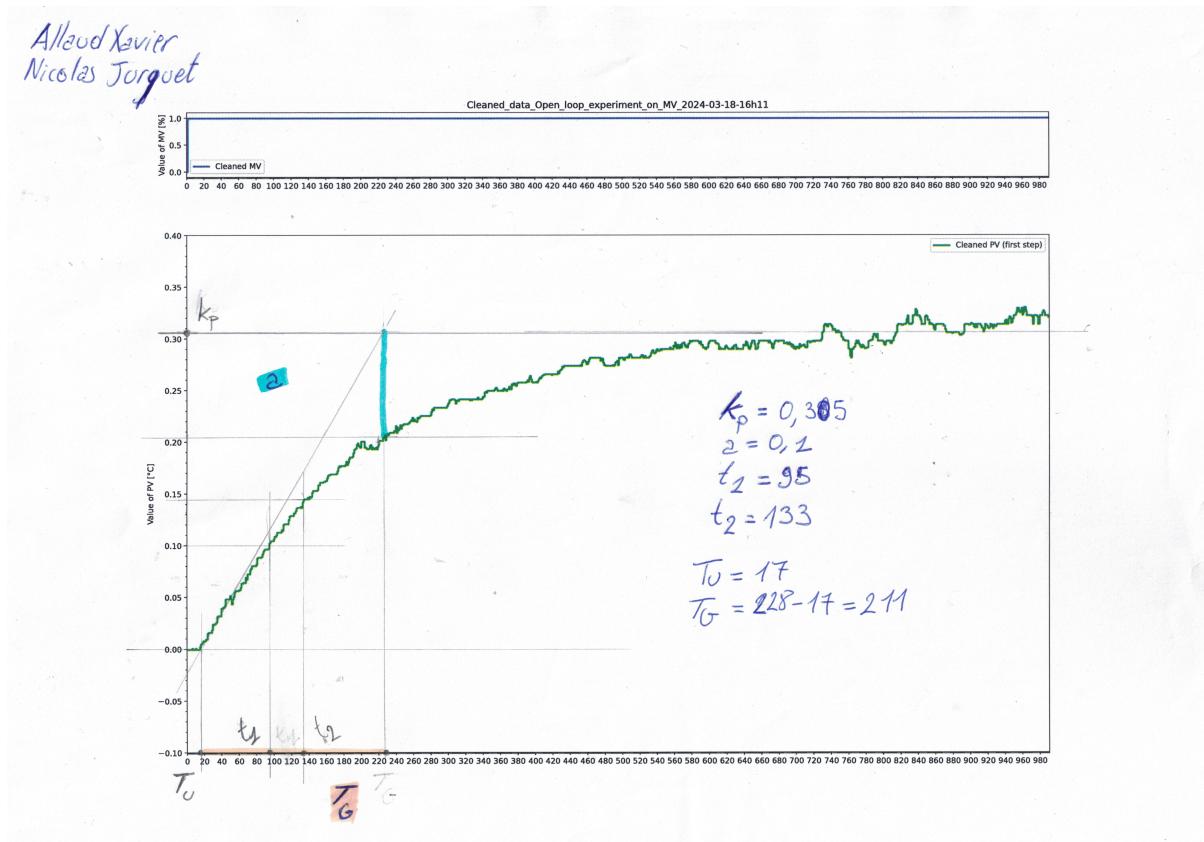


FIGURE 18 – Obtention des paramètres d'approximation pour un step sur MV