Commande avancée

Travaux Pratiques

Optimisation du comportement d'un système
par une méthode bio-inspirée

Julien Marot

1 Présentation du problème

Nous souhaitons dans ce TP minimiser deux critères de performance caractérisant un système d'ordre 2, corrigé par un correcteur proportionnel dérivé de gain K_c et de paramètre T_d.

Le système à corriger est le suivant :

$$G(p) = \frac{K}{(1+\tau_1 p)(1+\tau_2 p)} \tag{1.1}$$

Le système corrigé est caractérisé par un vecteur de paramètres $\mathbf{x} = [K_c, T_d]$. Ses performances se mesurent *via* son dépassement (quand il existe) et l'erreur par rapport à la valeur finale (supposée valeur en régime établi).

L'objectif est de trouver le couple de paramètres $\mathbf{x} = [K_c, T_d]$ qui minimise un critère concernant le dépassement ou l'erreur.

Nous observerons la réponse indicielle du système lorsque l'on minimise l'un OU l'autre de ces critères.

Pour ce faire, nous mettrons en oeuvre une méthode d'optimisation bio-inspirée.

2 Définition d'un critère

On suppose que le système que l'on étudie présente un dépassement.

On pose $J(\mathbf{x}) = |D(\mathbf{x}) - D_{objectif}|$, où $|.|$ est la valeur absolue. $D_{objectif}$ est un objectif à atteindre en termes de dépassement. λ_1 et λ_2 sont des coefficients de régularisation.

Notre but est de minimiser la fonction J, qui est non-linéaire, en fonction de $\mathbf{x}$, c'est-à-dire, trouver :

$$\widehat{\mathbf{x}} = \underset{\mathbf{x}}{argmin}(J(\mathbf{x})) \tag{2.1}$$

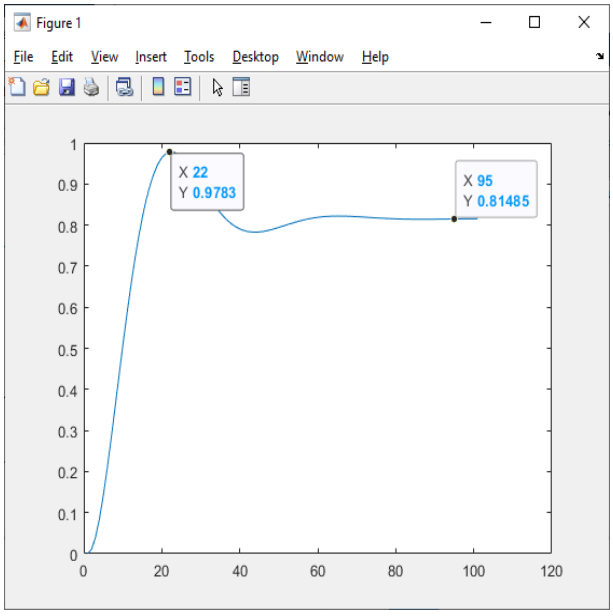

Figure 2.1 – Exemple de solution obtenue avec $K = 4$, $\tau_1 = 10, \tau_2 = 20$

La figure 2 montre la réponse indicielle obtenue quand on minimise J en cherchant la valeur optimale de K_c, lorsqu'un simple correcteur proportionnel est utilisé.

3 Principe de la méthode PSO

Il existe plusieurs méthodes d'optimisation qui permettent de retrouver le minimum global d'une fonction. On les appelle les méthodes d'optimisation globales : asymptotiquement, elle ne risquent pas de converger vers un minimum local de la fonction à minimiser. Elles sont donc exploitables pour optimiser des fonctions non-linéaires. La méthode DIRECT [1] suppose que la fonction à minimiser est k-lipschitzienne, ce qui n'est pas le cas du critère J. La méthode de Nelder-Mead [3] permet de minimiser une fonction quand elle présente plusieurs minima relatifs, mais seulement quand il n'y a qu'une seule inconnue. Nous choisissons donc d'adapter la méthode 'particle swarm optimization' [2] ou PSO (optimisation par essaim de particules). En voici son principe, pour une itération it. On calcule une position :

$$\mathbf{x}_q(it+1) = \mathbf{x}_q(it) + \mathbf{v}_q^{\mathbf{x}}(it+1) \tag{3.1}$$

avec une vitesse $\mathbf{v}_q^{\mathbf{x}}$ définie par :

$$\begin{aligned} \mathbf{v}_q^{\mathbf{x}}(it+1) = W\ \mathbf{v}_q^{\mathbf{x}}(it) + \gamma_{1q}\ r1_q(\mathbf{p}_q^{\mathbf{x}} - \mathbf{y}_q^{\mathbf{x}}(it)) \\ \ldots + \gamma_{2q}\ r2_q(\mathbf{G}^{\mathbf{x}} - \mathbf{y}_q^{\mathbf{x}}(it)) \end{aligned} \tag{3.2}$$

Dans les Eqs. (3.1) et (3.2), $\mathbf{v}_q^{\mathbf{x}}(it)$ est la vitesse de la particule q à l'itération it, W est une inertie, γ_{1q} et γ_{2q} des constantes d'accélération qui encouragent une recherche plutôt locale ou plutôt globale respectivement. $r1_q$ et $r2_q$ sont des nombres aléatoires, $\mathbf{p}_q^{\mathbf{x}}$ est la meilleure position trouvée pour la particule q, $\mathbf{G}^{\mathbf{x}}$ est la meilleure position pour l'ensemble des particules, et $\mathbf{y}_q^{\mathbf{x}}(it)$ est la position courante de la particule q à l'itération it. Quand le nombre maximum d'itérations choisi ($maxit$) est atteint, le vecteur position $\mathbf{x}(maxit)$ contient l'estimation finale de tous les paramètres dans $\widehat{\mathbf{x}}$.

4 Mise en oeuvre

4.1 Fonction Critère

Créez la fonction qui définit le système et un critère de performance en reproduisant et en complétant le code ci-dessous :

```
function Criterion=Compute_criterion_System(in)

K=4;
tau1=10; %%% sec.
tau=tau1;
tau2=tau1*2; %%% sec.

Kc=in(1);
Td=in(2);
Ti=in(3);%Inf;

p=tf('s');

Pterm= ??;

if Td>0
Dterm= ??
else
Dterm=0;
end

if isinf(Ti)~=1
Iterm= ?? ;
else
Iterm=0;
end
```

```
Cor=Kc * (Pterm + Dterm + Iterm);

num=K;
ordresys=2;
den=[ones(1,ordresys-2),conv([tau1,1],[tau2,1])];

BF=??;

Consigne=1;
Step_Response=Consigne*step(BF,tau*100);
[Max_Y,IndMax]=max(Step_Response);
Y_Final=Step_Response(end);

Error=??;
Error=Error(:);

%IAE: integrale(abs(e(t)))
IAE=??;

Criterion=IAE;

% Depassement_Objectif=20;
% Depassement=abs(Max_Y-Y_Final)/Y_Final * 100;
% CriterionD=abs(Depassement-Depassement_Objectif);
% Criterion=CriterionD;

figure(1)
plot(Step_Response)
```

On choisira un temps d'observation relativement long.
Question 1 : Une fois le code complété, faites tourner la fonction `Compute_criterion_System` avec quelques exemples pour les valeurs dans le vecteur `in`.
Par exemple : $K_c = 10$, $T_d = 5$, $T_i = 8$

4.2 Méthode d'optimisation

4.2.1 Construction de la fonction objectif

Créez la fonction critère, qui donne les valeurs du critère à minimiser pour plusieurs agents de recherche, en reproduisant et en complétant le code ci-dessous :

```
function [out]=Systeme(in)

Criterion=zeros(size(in,1),1);
for indpart=1:size(in,1)
Criterion(indpart)=Compute_criterion_System(in(indpart,:));
end

out=Criterion;

end
```

Question 2 : testez la fonction `Systeme` en donnant une matrice `in` à trois agents, *i.e.* trois couples de valeurs.

4.2.2 Paramétrage de PSO

Dans les programmes fournis par l'enseignant, vous devez mettre à jour le programme `Setup_PSO.m` :
Question 3 : Choisissez la fonction présentée en section 2. Il s'agit de la fonction 18.

Question 4 : Fixez la dimension du vecteur $\mathbf{x}$ à 2. Que représente cette dimension ?

Question 5 : Choisissez de minimiser l'écart entre le dépassement et un dépassement 'visé' comme 20% par exemple. Prenez un nombre de particules égal à 40 et un nombre d'itérations égal à 50. Faites tourner la méthode PSO.

Quelle est la valeur de $\widehat{\mathbf{x}}$ obtenue par la méthode ? Vérifiez si l'objectif en termes de dépassement, et celui en terme d'erreur sont atteints. Si ce n'est pas le cas, augmentez le nombre d'itérations.

Question 6 : On souhaite pénaliser les solutions qui ne présentent pas de dépassement. Proposez une solution pour cela.

Question 7 : Choisissez un autre critère à minimiser, comme l'IAE ('integral of absolute error'), définie par :

IAE : $\int |(e(t)|$

où $e(t)$ désigne l'erreur entre la sortie au temps t et la sortie finale. Quelle est l'allure de la sortie obtenue avec les paramètres optimaux ?

Question 8 : Modifiez les programmes pour que le correcteur soit un PID et non plus un proportionnel dérivé. Demandez à la méthode PSO d'estimer la valeur optimale de K, Td, Ti, selon les critères cités précédemment.

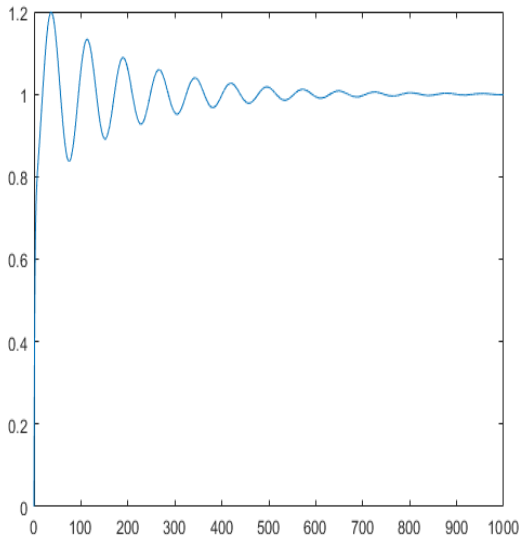

Figure 4.1 – Exemple de solution obtenue pour un dépassement de 20 %, avec un PID : $K = 0.4$, $Td = 56.4$, $Ti = 2$

5 Bilan, conclusion

Faites quelques expériences complémentaires : que pouvez-vous choisir d'autre comme critère ? Combinez deux types de critères contradictoires, comme par exemple un dépassement élevé et une erreur faible.
Changez l'ordre du système. Avec un système d'ordre plus élevé, peut-on encore obtenir la valeur de dépassement voulue ?

Faites quelques commentaires généraux.

Bibliographie

[JPS93] D.R. JONES, C.D. PERTUNEN et B.E. STUCKMAN. « Lipschitzian optimization without the Lipschitz constant ». In : *Journal of Optimization and Applications* 79 (1993), p. 157-181 (cf. p. 4).

[KE95] J. KENNEDY et R. EBERHART. « Particle swarm optimization ». In : *IEEE International Conference on Neural Networks*. Perth, 1995, p. 1942-1948 (cf. p. 4).

[Lag+98] Jeffrey C LAGARIAS, James A REEDS, Margaret H WRIGHT et al. « Convergence properties of the Nelder–Mead simplex method in low dimensions ». In : *SIAM Journal on optimization* 9.1 (1998), p. 112-147 (cf. p. 4).