Déconvolution de trains d'impulsions. Application au controle non destructif par ultrasons

Yassine Jamoud, Samy Haffoudhi

25 décembre 2021

Introduction

Dans ce TP, l'objectif est d'analyser un objet à l'aide de signaux ultrasons. La séquence de réflectivité x sera un singal parcimonieux où chaque pics correspond à un changement d'impédances. L'objectif est de détecter ces pics efficacement. Pour se faire, on considère que notre signal reçu y, est la convolution du singal recherché x avec la réponse impulsionelle h du transducteurs avec une incertitude comportant entre autre le bruit de mesure et les erreurs de modélisation. On adopte donc le modèle :

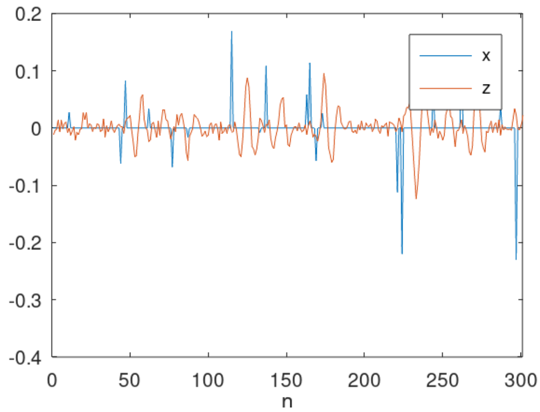
$$y = x * h + incertitudes$$

1 Modèle Direct

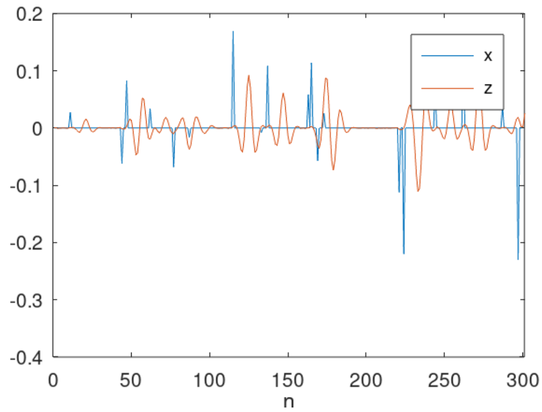
Simulons le problème en utilisant un exemple de réponse impulsionnelle. On adopte ici le modèle matriciel

$$z = Hx + b$$

Pour simuler le problème efficacement on utilise la convolution de matlab, qui nous évite d'utiliser de grosse matrice dans l'algorithme. On peut alors générer un signal z avec un rapport signal sur bruit donné.



(a) Simulation d'un problème direct avec $RSB_{dB} = 10dB$



(b) Simulation d'un problème direct avec $RSB_{dB} = 40dB$

2 Déconvolution par pénalisation l_1

1. Commençons par utliser naïvement la solution des moindres carrés $\hat{x_{MC}} = (H^T H)^{-1} H^T z$.

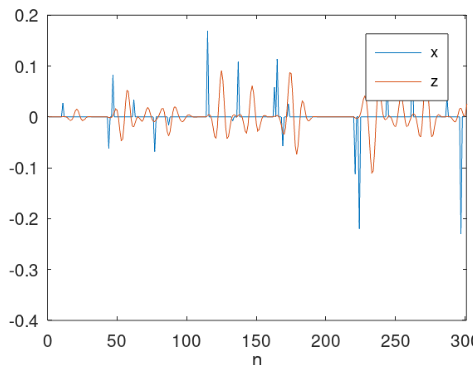


FIGURE 2 – Simulation avec un rapport signal sur bruit infini

Pour un rapport signal sur bruit infini, le bruit est quasiment inexistant. La solution des moindres carrés fonctionne donc bien :

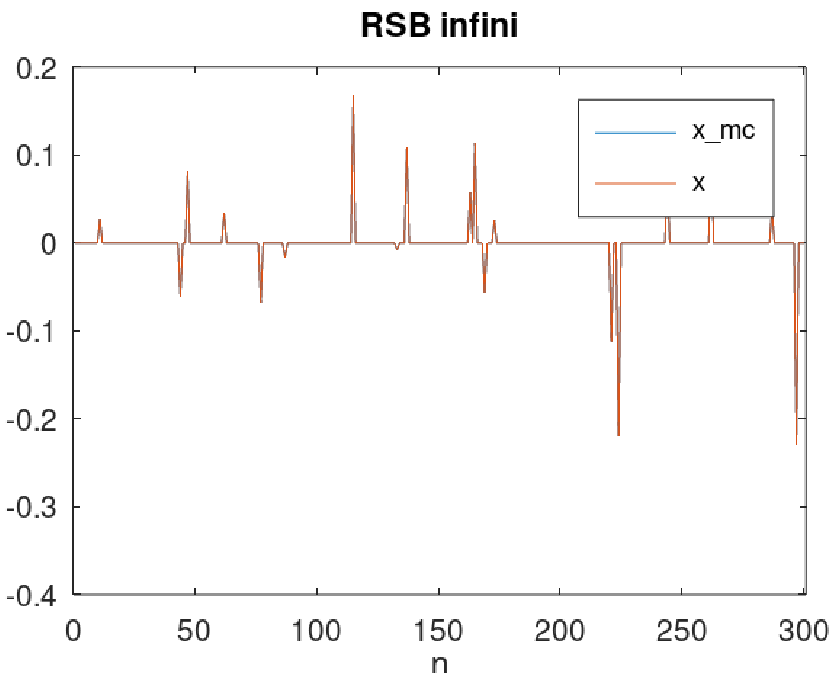


FIGURE 3 – Solution des moindres carrés avec un rapport signal sur bruit infini

On retrouve effectivement un signal z identique au signal x.

2. Utilisons maintenant un RSB_{dB} très bon.

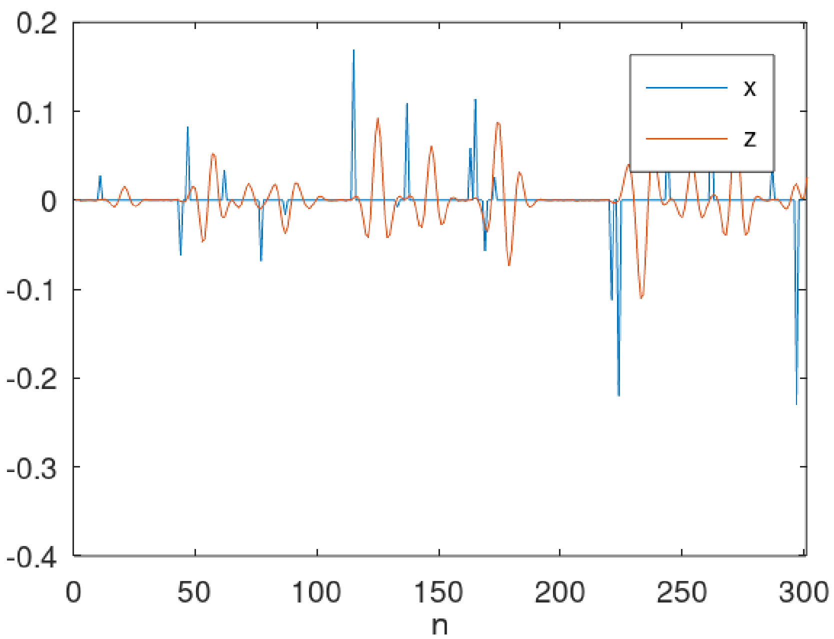


FIGURE 4 – Simulation avec un rapport signal sur bruit de $40dB$

Cette fois-ci la solution des moindres carrés est catastrophique :

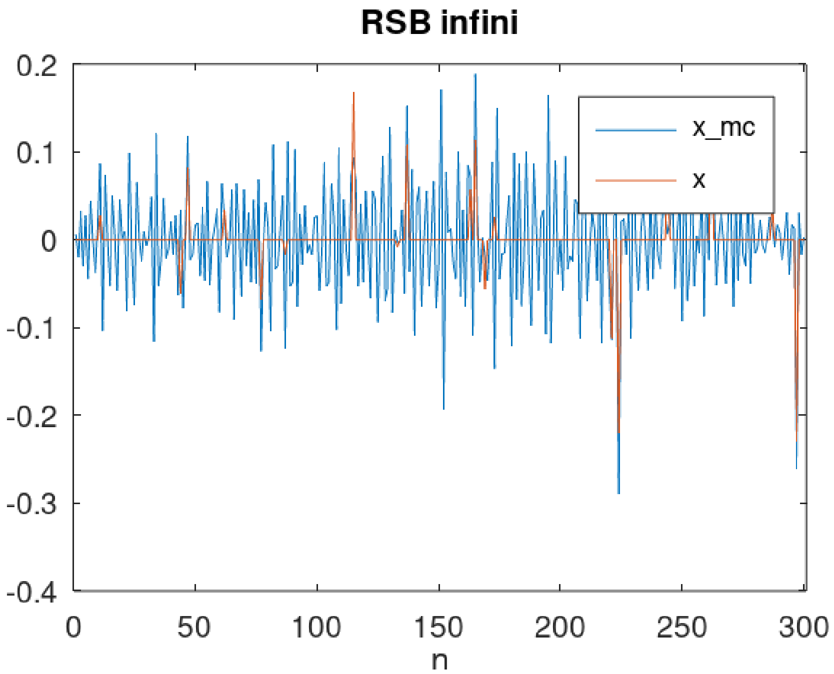


FIGURE 5 – Solution des moindres carrés avec un rapport signal sur bruit de $40dB$

En effet, $z = Hx + b$ et $\hat{x_{MC}} = (H^T H)^{-1} H^T z$.
Si on injecte z on retrouve :

$$\hat{x_{MC}} = (H^T H)^{-1} H^T (Hx + b) = x + (H^T H)^{-1} H^T b$$

Le bruit est donc multiplier par un facteur très grand devant lui. En effet, H étant mal conditionnée, les valeurs propres de $H^T H$ sont très petite, donc l'inverse fournit de très grande valeurs propres.

3. On pénalise donc la solution des moindres carrés par norme l_1 :

$$\hat{x} \ \ minimise \ \ J(x) = \frac{1}{2}||z - Hx||^2 + \mu \sum_{m=1}^{M} |x_m|$$

On utilise l'algorithme $ISTA$ car la non-differentiabilité en 0 ne permet pas d'avoir de solution analytique.

4. C est la plus grande valeur propre de de $H^T H$. On utilise donc $max(eig(H^T H))$ pour obtenir une valeur.
5. Évaluons les performances de cette algorithme. Empiriquement, on choisit $\mu = 0.01$. On retrouve alors les resultats suivant :

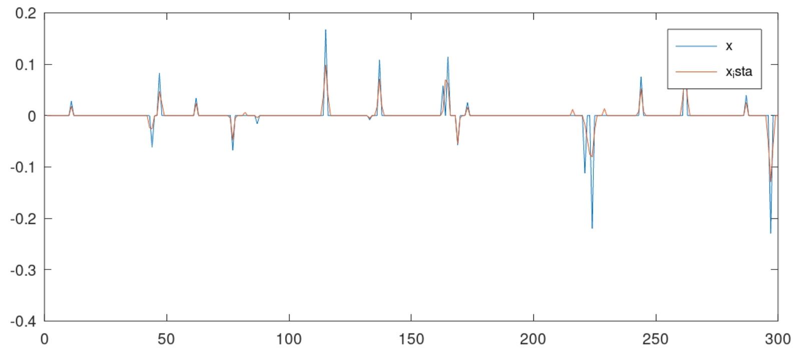


FIGURE 6 – Solution de l'algorithme ISTA avec un rapport signal sur bruit de $40dB$

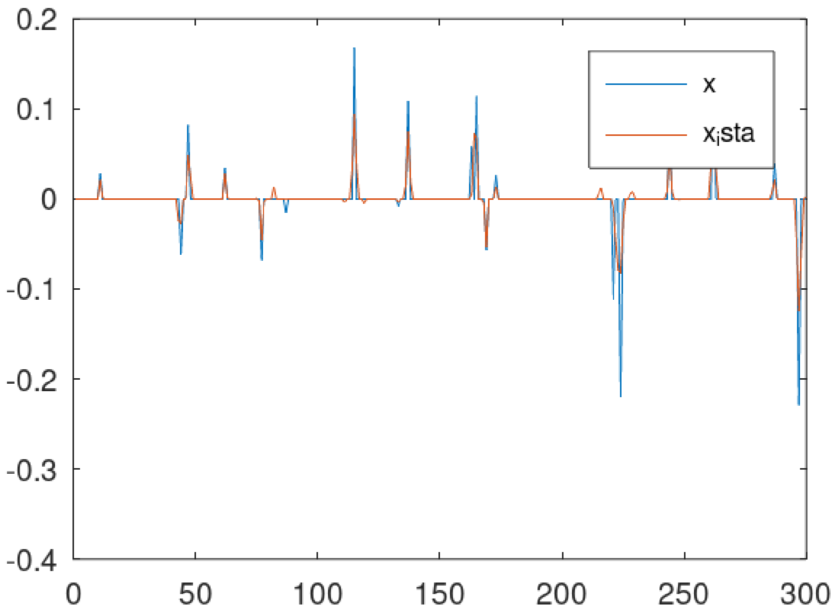


FIGURE 7 – Solution de l'algorithme ISTA avec un rapport signal sur bruit de $20dB$

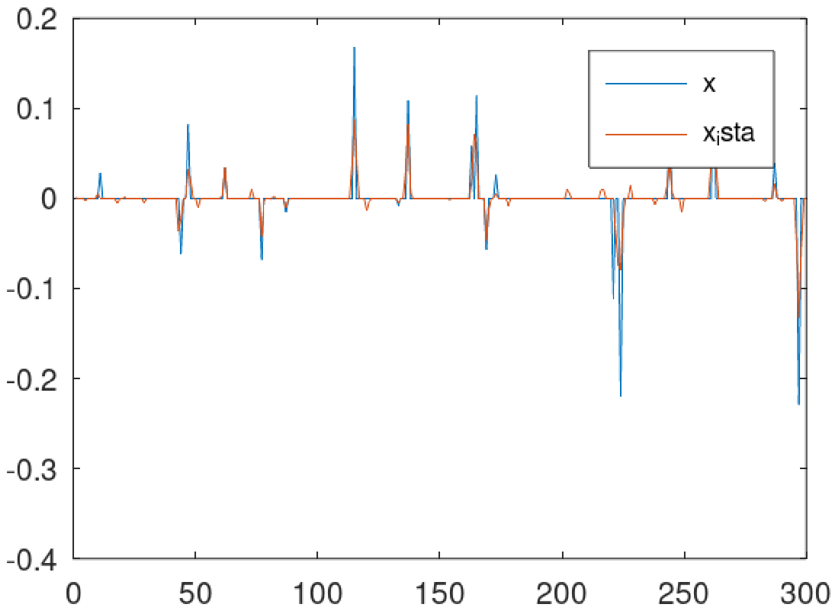


FIGURE 8 – Solution de l'algorithme ISTA avec un rapport signal sur bruit de $10dB$

Pour une presence de bruit presque nul ou très bonne, on arrive à obtenir un resultat très correct. Seul les pics vraiment proche posent un problèmes car ils sont associés comme un seul pics. Pour un rapport singal sur bruit moyen, on commence à voir apparaître des fausses valeurs due au bruit.

3 Mesures d'épaisseurs de plaques par ultrasons

Mettons en oeuvre nos solution sur des signaux réels

1. Pour le signal z_1, les pics sont assez éloignés pour mesurer l'epaisseur de la plaque sans déconvoluer le signal. L'avantage de ce genre de signal est de pouvoir extraire h, la réponse impulsionnelle très facilement en récupérant le bon nombre de points.

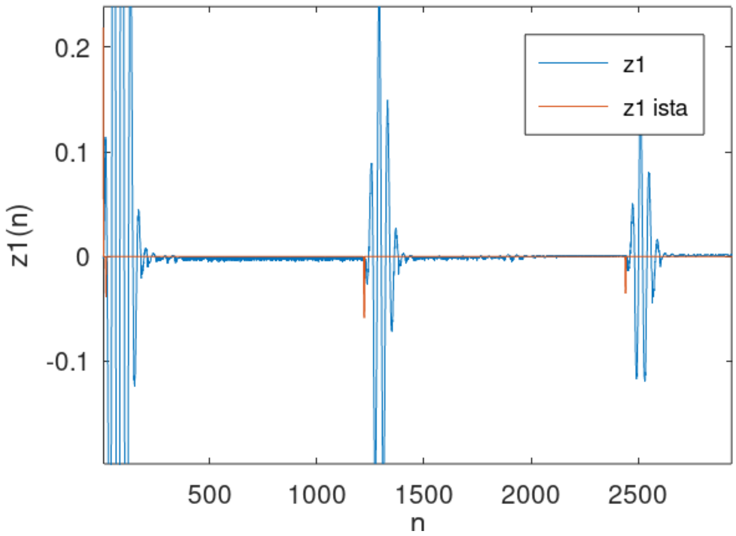


FIGURE 9 – Déconvolution du signal z_1 avec $\mu = 1$

On retrouve un intervalle temporelle entre deux pics d'environs 1206 points. La fréquence d'échantillonnage étant de $100MHz$, cela correspond à un intervalle de $\delta t = 1.2 * 10^{-5}s$. La vitesse du son dans l'aluminium étant de $v = 6380m.s^{-1}$, la distance parcourue est donc $d = \delta t * v = 0.077m$. Ainsi l'epaisseur de la plaque est de $3.85cm$.

2. Pour une plaque plus fine, on retrouve la deconvolution suivante :

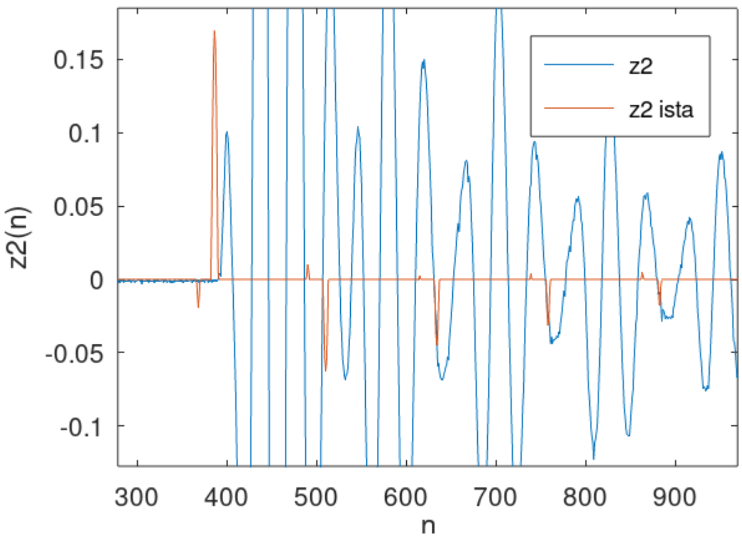


FIGURE 10 – Déconvolution du signal z_1 avec $\mu = 1$

Encore une fois, on peut identifier identiquement l'intervalle entre deux pics. On retrouve environ 123 points soit $1.23 * 10^{-6}s$. L'épaisseur de la plaque est donc de $0.4cm$.

Conclusion

Ce TP à ainsi permis de mettre en évidence l'exemple d'utilisation des algorithmes de deconvolution. On a pu mettre en lumière l'inefficacité des moindres carrés en présence de bruit. La régularisation est donc nécessaire. Pour la norme l_1 la non-differentiabilité nous force à utiliser des algorithmes pour approximer au mieux le min de J. Malgrés sa sensibilité aux bruits importants, cet algorithme c'est révéler efficace dans la détection des piques par ultrasons. La limite pourra être atteinte pour des materiaux très fin et donc des pics trop proche.

A Simulation d'un problème direct

```
function z = probleme_direct(x, h, RSB)
    N = length(x);
    y = conv(x, h, "full");
    E_signal = sum(y.^2);
    var_bruit = E_signal*10^(-RSB/10)/N;
    z = y + sqrt(var_bruit)*randn(size(y));
end
```

B Seuillage doux

```
function y = seuillage_doux(z, mu)
    y = zeros(length(z),1);
    for i = 1:length(z)
        if z(i) >= mu
            y(i) = z(i) - mu;
        elseif z(i) <= -mu
            y(i) = z(i) + mu;
        end
    end
end
```

C Algorithme ISTA

```
function x = ista(z, h, C, mu, max_iter, x0)
    for q = 1:max_iter
        x0 = seuillage_doux(x0 + 1/C*conv(z - conv(x0, h, 'full'), flip(h), 'valid')
    ,mu/C);
    end
    x = x0;
end
```

D Code utilisation des méthodes

```
close all
clear all

load('dataCND.mat')

z = probleme_direct(x, h, inf);

figure(1)
plot(x);
hold on;
plot(z);
xlabel('n')
legend('x', 'z')
xlim([0 length(x)+1])

% solution moindres carres

z = probleme_direct(x, h, inf);

x_mc = linsolve(H'*H,H'*z);
figure(2)
plot(x_mc)
```

```
hold on
plot(x)
xlabel('n')
legend('x\_mc', 'x')
xlim([0 length(x)+1])
title('RSB infini')

z = probleme_direct(x, h, 40);

x_mc = linsolve(H'*H,H'*z);
figure(3)
plot(x_mc)
hold on
plot(x)
xlabel('n')
legend('x\_mc', 'x')
xlim([0 length(x)+1])
title('RSB infini')

% ista

max_iter = 100;
C =  max(eig(H'*H));
mu = 0.01;
z = probleme_direct(x, h, 10);
x_ista = ista(z, h, C, mu, max_iter, zeros(length(z)-length(h)+1,1));
figure(4)
plot(x)
hold on
plot(x_ista)
legend('x', 'x_ista')


load('dataplaques.mat')
figure(5)
plot(z1)

h = z1(1:250);
C = max(abs(fft(h,8*2^nextpow2(length(h)))).^2);
mu = 1;
max_iter = 1000;
x_ista1 = ista(z1, h, C, mu, max_iter, zeros(length(z1)-length(h)+1,1));

figure(6)
plot(z1)
hold on
plot(x_ista1)
legend('z1', 'z1 ista')
xlabel('n')
ylabel('z1(n)')

mu = 1;
max_iter = 1000;
%x_ista2 = ista(z2, h, C, mu, max_iter, zeros(length(z2)-length(h)+1,1));

figure(7)
plot(z2)
hold on
%plot(x_ista2)
```

```
legend('z2', 'z2 ista')
xlabel('n')
ylabel('z2(n)')
```