16 Introduction au filtrage adaptatif

Le filtrage adaptatif est basé sur la recherche de paramètres optimaux par minimisation d'un critère de performance. Fréquemment, cette minimisation se fait en recherchant les moindres carrés.

Étant donné le cadre dans lequel cette présentation est faite, on commencera par rappeler quelques définitions d'estimateurs statistiques puis on montrera ce que sont la régression linéaire et le filtrage de Wiener avant de parler du filtrage adaptatif proprement dit.

16.1 Notions de probabilités

16.1.1 Définitions de quelques estimateurs statistiques

Considérant une variable aléatoire réelle z, on la caractérise généralement à l'aide des grandeurs suivantes :

1. Sa **valeur moyenne** :
$$\mu_z = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} z(n) \tag{16.1}$$
On notera que la valeur moyenne μ_z représente la composante continue du signal autour de laquelle prennent place les fluctuations.
2. Sa **puissance moyenne** :
$$P_z = \mu_{z^2} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} z^2(n) \tag{16.2}$$
3. Sa **variance** qui mesure la puissance des fluctuations autour de la valeur moyenne
$$\begin{aligned} \sigma_z^2 &= \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \left(z(n) - \mu_z\right)^2 \\ &= \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \left(z^2(n) - 2\mu_z\, z(n) + \mu_z^2\right) \\ &= \mu_{z^2} - 2\mu_z\,\mu_z + \mu_z^2 = \mu_{z^2} - 2\mu_z^2 + \mu_z^2 \end{aligned}$$
qui vaut finalement
$$\sigma_z^2 = \mu_{z^2} - \mu_z^2 \tag{16.3}$$

4. Son **écart-type** (ou déviation standard) défini comme la racine carrée de la variance :
$$\sigma_z = \sqrt{\mu_{z^2} - \mu_z^2} \tag{16.4}$$
Sa valeur est égale à la valeur efficace des variations du signal autour de la valeur moyenne.

Il est intéressant de noter que la puissance moyenne (μ_{z^2}) de la variable z peut également s'écrire sous la forme

$$\mu_{z^2} = \mu_z^2 + \sigma_z^2 \tag{16.5}$$

On voit ainsi que la puissance de la variable μ_{z^2} est égale à la puissance de sa valeur moyenne μ_z^2 plus la puissance de ses fluctuations σ_z^2.

16.1.2 Remarques

1. Il est intéressant de relever que si l'on considère une notation vectorielle du type :
$$z = [z(0), \ldots, z(N-1)], \qquad z^T = \begin{bmatrix} z(0) \\ \vdots \\ z(N-1) \end{bmatrix}$$
la puissance s'écrit simplement sous la forme d'un produit scalaire :
$$P_z = \frac{1}{N} \sum_{n=0}^{N-1} z^2(n) = \frac{1}{N}\, z\, z^T \tag{16.6}$$

2. De même, l'indépendance (ou la non correspondance) de deux signaux ou vecteurs peut se mesurer avec le produit scalaire :
$$r = x\, y^T = \sum_{n=0}^{N-1} x(n)\, y(n) \tag{16.7}$$
Dans le cas où les signaux sont orthogonaux (c'est à dire indépendants), $x\, y^T$ sera nul alors que si les signaux sont fortement dépendants (ou ressemblants), la valeur du produit $x\, y^T$ sera proche de son maximum.

16.1.3 Fonction de répartition et densité de probabilités

Dans le cas d'une variable aléatoire continue x, on définit la probabilité d'avoir la valeur mesurée x' inférieure à une valeur x donnée

$$P(x) \equiv \text{prob}(x' < x) \tag{16.8}$$

Cette fonction porte le nom de fonction de répartition la variable x.

La probabilité d'avoir la valeur mesurée x' comprise entre deux valeurs x et $x + \Delta x$ vaut donc

$$P(x < x' < x + \Delta x) = P(x' < x + \Delta x) - P(x' < x) \tag{16.9}$$

Cette relation permet de définir la densité de probabilité

$$p(x) \equiv \lim_{\Delta x \to 0} \frac{P(x < x' < x + \Delta x)}{\Delta x} = \frac{dP(x)}{dx} \tag{16.10}$$

et d'en tirer la fonction de répartition

$$P(x) = \int_{-\infty}^{x} p(x)\, dx \tag{16.11}$$

avec comme propriété évidente

$$P(-\infty < x' < +\infty) = P(\infty) = 1 \tag{16.12}$$

C'est la densité de probabilité qui est généralement utilisée comme modèle de pour décrire la répartion des valeurs d'une variable aléatoire. À partir de celle-ci, on peut calculer la valeur moyenne, la variance et la puissance d'une variable x à l'aide de

$$\mu_x = \int_{-\infty}^{+\infty} x\, p(x)\, dx \tag{16.13}$$

$$\sigma_x^2 = \int_{-\infty}^{+\infty} (x - \mu_x)^2\, p(x)\, dx \tag{16.14}$$

$$\mu_{x^2} = \int_{-\infty}^{+\infty} x^2\, p(x)\, dx \tag{16.15}$$

16.1.4 Modèles statistiques

Les modèles les plus fréquemment utilisés sont

1. La **répartition uniform**e entre deux valeurs extrêmes x_{min} et x_{max}

$$p(x) = \text{constante} = \frac{1}{x_{max} - x_{min}} = \frac{1}{\Delta x} \tag{16.16}$$

   On montre que dans ce cas, la variance vaut

$$\sigma_x^2 = \frac{(x_{max} - x_{min})^2}{12} = \frac{\Delta x^2}{12} \tag{16.17}$$

2. La **répartition gaussienne** entre $-\infty$ et $+\infty$

$$p(x) = \frac{1}{\sqrt{2\pi}\,\sigma_x} \exp\left(-\frac{(x - \mu_x)^2}{2\,\sigma_x^2}\right) \tag{16.18}$$

   Il est intéressant de relever que que la probabilité de trouver à l'intérieur des domaines $\pm\sigma_x$ et $\pm 3\sigma_x$ par rapport à la valeur moyenne valent respectivement

$$P\left(|x - \mu_x| < \sigma_x\right) = 68\% \tag{16.19}$$

$$P\left(|x - \mu_x| < 3\sigma_x\right) = 99.7\% \tag{16.20}$$

Suivant les applications, on peut imaginer d'autres distributions comme par exemple la **répartition exponentielle** décrite par

$$p(x) = \frac{1}{\sqrt{2}\,\sigma_x} \exp\left(-\sqrt{2}\,\frac{|x - \mu_x|}{\sigma_x}\right) \tag{16.21}$$

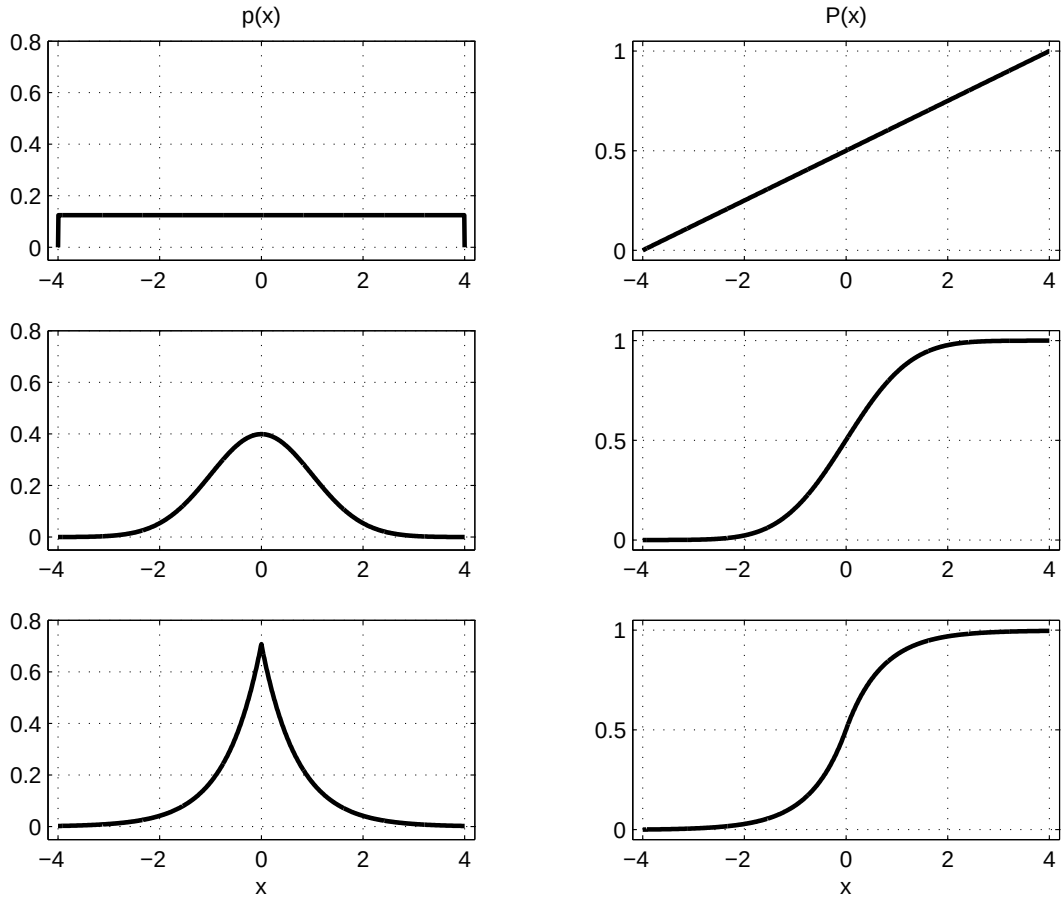

Fig. 16.1: Illustration des distributions uniforme, gaussienne et exponentielle

16.2 Régression linéaire

La régression linéaire consiste en la recherche de la droite passant au mieux parmi un ensemble de points mesurés (figure 16.2). Le critère conduisant à cet optimum est la minimisation des distances quadratiques entre les points mesurés et la droite optimum.

On notera que la régression linéaire s'applique aux systèmes statiques alors que l'approche de Wiener (que l'on verra dans la section suivante) sert à optimiser des systèmes évoluant au cours du temps.

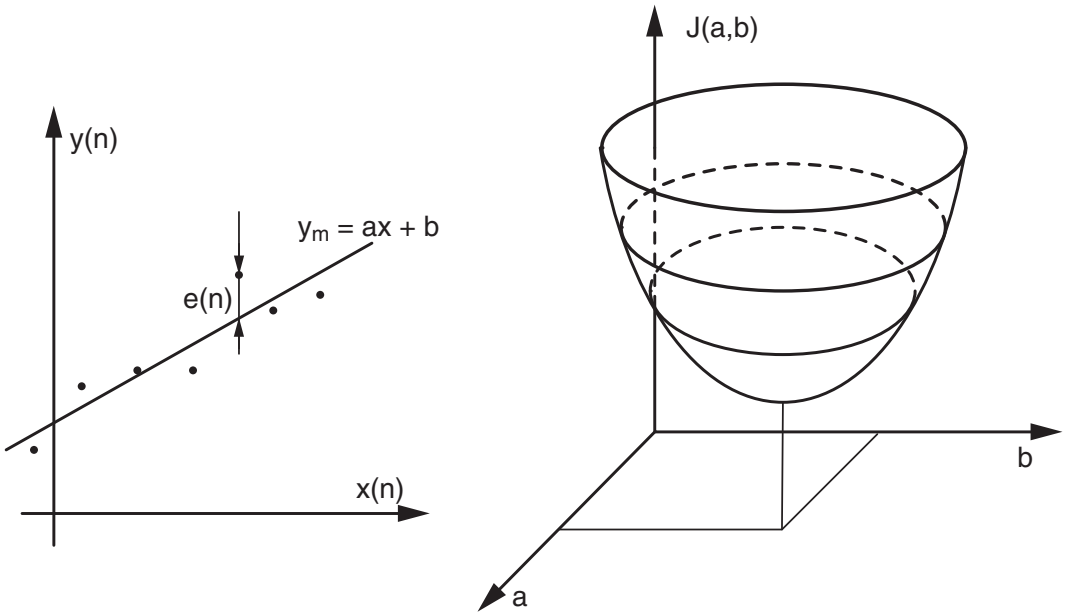

FIG. 16.2: Régression linéaire

16.2.1 Mesure, modèle et écart

Comme on souhaite faire passer une droite parmi un ensemble de points,on se donne un modèle dont l'équation est :

$$y_m = ax + b \tag{16.22}$$

L'écart de $y(n)$ par rapport au modèle s'écrit donc :

$$e(n) = y(n) - y_m(n)$$

$$e(n) = y(n) - (a\,x(n) + b) \tag{16.23}$$

Si l'on décrit la mesure $y(n)$ par rapport au modèle $y_m(n)$, on a évidemment :

$$y(n) = y_m(n) + e(n) \tag{16.24}$$

On associe généralement deux grandeurs à l'écart $e(n)$:

1. sa valeur moyenne μ_e qui doit tendre vers 0 si le modèle n'est pas biaisé ;
2. sa puissance σ_e^2 qui doit diminuer lorsque a et b se rapprochent des "vraies" valeurs liant y à x.

On notera que pour le calcul d'une régression linéaire, on fait l'hypothèse qu'il n'y a pas de bruit sur la valeur de la variable indépendante $x(n)$.

16.2.2 Minimisation de l'écart quadratique

L'obtention de la droite passant au mieux parmi les points mesurés nécessite la recherche du minimum d'une fonction dépendant des paramètres recherchés a et b. Pour cela, on définit un critère d'optimisation qui mesure la puissance ou la variance de l'écart :

$$J(a,b) = \frac{1}{N}\sum_{n=0}^{N-1} e^2(n) = \frac{1}{N}\sum_{n=0}^{N-1} \left(y(n) - (ax(n)+b)\right)^2 \tag{16.25}$$

Lorsque l'écart quadratique est minimum, on a :

$$\frac{\partial J(a,b)}{\partial a} = 0 \qquad\qquad \frac{\partial J(a,b)}{\partial b} = 0 \tag{16.26}$$

avec

$$\begin{aligned}
\frac{\partial J(a,b)}{\partial a} &= \frac{1}{N}\sum_{n=0}^{N-1} 2\,\left(y(n) - (ax(n)+b)\right)\,(0 - x(n) - 0) \\
&= \frac{2}{N}\sum_{n=0}^{N-1} \Big(-x(n)\,y(n) + a\,x^2(n) + b\,x(n)\Big) \\
&= \frac{2}{N}\left(-\sum_{n=0}^{N-1} x(n)\,y(n) + a\sum_{n=0}^{N-1} x^2(n) + b\sum_{n=0}^{N-1} x(n)\right)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial J(a,b)}{\partial b} &= \frac{1}{N}\sum_{n=0}^{N-1} 2\,\left(y(n) - (ax(n)+b)\right)\,(0 - 0 - 1) \\
&= \frac{2}{N}\sum_{n=0}^{N-1} \left(-y(n) + ax(n) + b\right) \\
&= \frac{2}{N}\left(-\sum_{n=0}^{N-1} y(n) + a\sum_{n=0}^{N-1} x(n) + \sum_{n=0}^{N-1} b\right)
\end{aligned}$$

On en tire 2 équations dont les inconnues sont a et b :

$$a\,\frac{1}{N}\sum_{n=0}^{N-1} x^2(n) + b\,\frac{1}{N}\sum_{n=0}^{N-1} x(n) \;=\; \frac{1}{N}\sum_{n=0}^{N-1} x(n)\,y(n) \tag{16.27}$$

$$a\,\frac{1}{N}\sum_{n=0}^{N-1} x(n) + \frac{1}{N}\sum_{n=0}^{N-1} b \;=\; \frac{1}{N}\sum_{n=0}^{N-1} y(n) \tag{16.28}$$

16.2.3 Équations de la régression linéaire

Se souvenant de la définition d'une valeur moyenne, on voit que les équations (16.28) et (16.27) s'écrivent plus simplement sous la forme :

$$a\,\mu_x + b = \mu_y \tag{16.29}$$

$$a\,\mu_{x^2} + b\,\mu_x = \mu_{xy} \tag{16.30}$$

Sous forme matricielle, cela donne :

$$\begin{pmatrix} \mu_x & 1 \\ \mu_{x^2} & \mu_x \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \mu_y \\ \mu_{xy} \end{pmatrix}$$

dont la solution est

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \mu_x & 1 \\ \mu_{x^2} & \mu_x \end{pmatrix}^{-1} \begin{pmatrix} \mu_y \\ \mu_{xy} \end{pmatrix} \tag{16.31}$$

L'inversion de la matrice et le calcul explcite de a et b donnent alors

$$a = \frac{\mu_x\,\mu_y - \mu_{xy}}{\mu_x^2 - \mu_{x^2}} \tag{16.32}$$

$$b = \frac{\mu_x\,\mu_{xy} - \mu_y\,\mu_{x^2}}{\mu_x^2 - \mu_{x^2}} \tag{16.33}$$

Dans le cas particulier où la droite passe par l'origine, les valeurs moyennes μ_x et μ_y sont nulles et on a :

$$a = \frac{\mu_{xy}}{\mu_{x^2}} = \frac{\sum x(n)\,y(n)}{\sum x^2(n)} = \frac{x^T\,y}{x^T\,x}, \qquad b = 0 \tag{16.34}$$

16.3 Filtrage de Wiener

Dans de nombreuses applications, les signaux temporels sont entachées d'une interférence ou d'un bruit non désirés. Il faut alors trouver une solution permettant de supprimer ou tout au moins réduire ces composantes perturbatrices. Dans le cas où le spectre du signal désiré et celui du signal perturbateur se superposent, il n'est pas possible de recourir au filtrage classique. Le filtre de Wiener apporte une solution à ce problème lorsque le processus est stationnaire.

16.3.1 Définition du problème

On considère ici le schéma de la figure 16.3 dans lequel on trouve :

1. le signal d'excitation $x(n)$ connu ou mesuré ;
2. le signal de sortie du processus $y_p(n)$ inatteignable ;

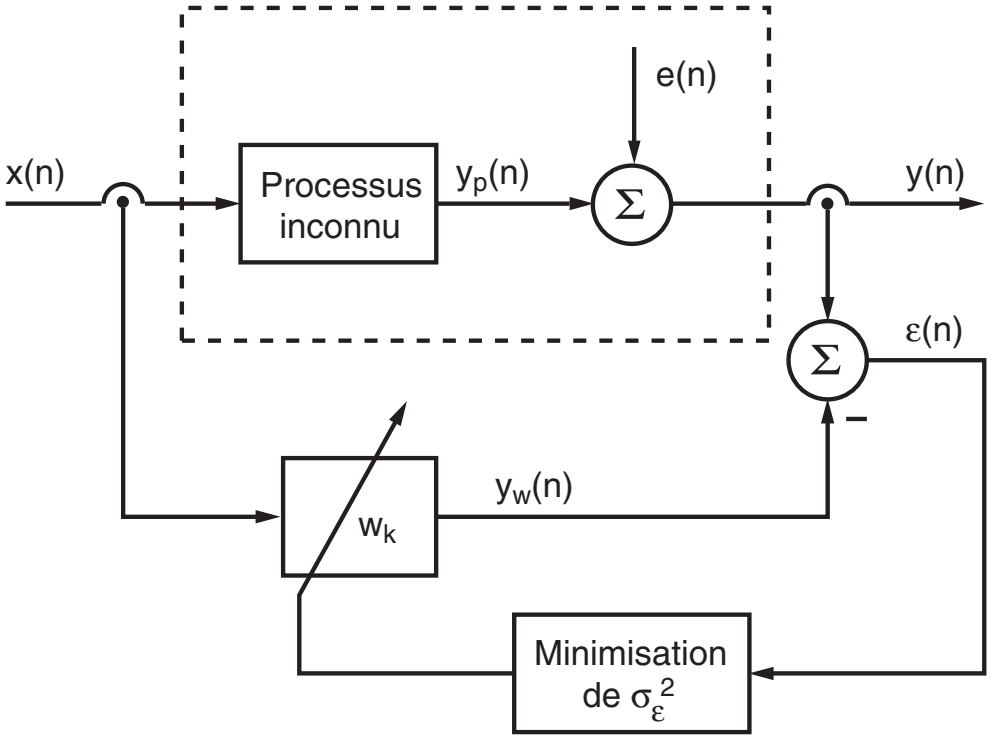

Fig. 16.3: Filtrage de Wiener

3. le signal de sortie mesuré $y(n)$ entâché d'une perturbation inconnue $e(n)$;
4. le signal modélisé $y_w(n)$ à l'aide des paramètres w_k ;
5. le signal d'écart $\epsilon(n)$ entre le modèle $y_w(n)$ et la mesure $y(n)$.

On admet que le signal mesuré $y(n)$, causé par l'excitation $x(n)$, peut être modélisé à l'aide d'un modèle MA (*Moving Average* = moyenne glissante) d'ordre p représentant un processus stationnaire inconnu :

$$y_p(n) = \sum_{k=0}^{p-1} w_k \, x(n-k)$$

Le but poursuivi est de trouver les coefficients w_k du modèle MA à partir de la mesure des signaux d'entrée $x(n)$ et de sortie $y(n)$.

La recherche d'une solution consiste à rendre $y_w(n)$ aussi proche que possible du signal $y_p(n)$ en minimisant l'erreur quadratique moyenne (*Mean Square Error* = MSE) par ajustage des coefficients w_k. Il est important de bien comprendre que

> ***si la solution exacte est trouvée, le signal d'écart*** $\varepsilon(n)$ ***n'est pas nul, mais égal à la peturbation*** $e(n)$ ***de la mesure.***

Afin d'alléger l'écriture de ce qui suit, on se contentera de traiter le cas particulier où le processus est décrit par 3 paramètres (l'extension à une dimension plus grande se fait sans difficulté)

$$W = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \end{pmatrix} \tag{16.35}$$

Dans ce cas, l'estimateur $y_w(n)$ du signal $y_p(n)$ vaut :

$$y_w(n) = w_0\, x(n) + w_1\, x(n-1) + w_2\, x(n-2) \qquad 0 \leq n \leq N-1 \tag{16.36}$$

16.3.2 Résolution au sens des moindres carrés

Le problème ainsi posé est proche de celui de la régression linéaire que l'on a étudié pour les systèmes statiques (ou sans mémoire). Dans le cas des systèmes dynamiques, les signaux évoluent temporellement. L'erreur est alors une fonction du temps que l'on cherche à réduire en minimisant sa valeur quadratique moyenne ; cela se fait en variant les coefficients inconnus w_k. On pose donc :

$$\varepsilon(n) = y(n) - y_w(n) \tag{16.37}$$

$$J = \frac{1}{N} \sum_{n=0}^{N-1} \left(y(n) - y_w(n)\right)^2 \tag{16.38}$$

Tenant compte de l'équation (16.36), il vient

$$J\left(w_0, w_1, w_2\right) = \frac{1}{N} \sum_{n=0}^{N-1} \left(y(n) - w_0\, x(n) - w_1\, x(n-1) - w_2\, x(n-2)\right)^2 \tag{16.39}$$

Le calcul des dérivées partielles de $J\left(w_0, w_1, w_2\right)$ par rapport à chacun des coefficients inconnus w_k donne

$$\begin{aligned}
\frac{\partial J}{\partial w_0} &= \frac{2}{N} \sum_{n=0}^{N-1} \left(y(n) - w_0\, x(n) - w_1\, x(n-1) - w_2\, x(n-2)\right)\, \left(-x(n)\right) \\
&= -\frac{2}{N} \sum_{n=0}^{N-1} \left(x(n)y(n) - w_0\, x(n)x(n) - w_1\, x(n)x(n-1) - w_2\, x(n)x(n-2)\right)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial J}{\partial w_1} &= \frac{2}{N} \sum_{n=0}^{N-1} \left(y(n) - w_0\, x(n) - w_1\, x(n-1) - w_2\, x(n-2)\right)\, \left(-x(n-1)\right) \\
&= -\frac{2}{N} \sum_{n=0}^{N-1} \left(x(n-1)y(n) - w_0\, x(n-1)x(n) - w_1\, x(n-1)x(n-1) - w_2\, x(n-1)x(n-2)\right)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial J}{\partial w_2} &= \frac{2}{N} \sum_{n=0}^{N-1} \left(y(n) - w_0\, x(n) - w_1\, x(n-1) - w_2\, x(n-2)\right)\, \left(-x(n-2)\right) \\
&= -\frac{2}{N} \sum_{n=0}^{N-1} \left(x(n-2)y(n) - w_0\, x(n-2)x(n) - w_1\, x(n-2)x(n-1) - w_2\, x(n-2)x(n-2)\right)
\end{aligned}$$

Tenant compte de la définition de la fonction de corrélation

$$r_{xy}(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n)\, y(n+k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n-k)\, y(n) = r_{yx}(-k) \tag{16.40}$$

on voit que ces trois dérivées s'écrivent plus simplement sous la forme

$$\begin{aligned}
\frac{\partial J}{\partial w_0} &= -2\left(r_{xy}(0) - w_0 r_{xx}(0) - w_1 r_{xx}(-1) - w_2 r_{xx}(-2)\right) \\
\frac{\partial J}{\partial w_1} &= -2\left(r_{xy}(+1) - w_0 r_{xx}(+1) - w_1 r_{xx}(0) - w_2 r_{xx}(-1)\right) \\
\frac{\partial J}{\partial w_2} &= -2\left(r_{xy}(+2) - w_0 r_{xx}(+2) - w_1 r_{xx}(+1) - w_2 r_{xx}(0)\right)
\end{aligned}$$

Comme l'erreur quadratique obtenue est minimum lorsque ces dérivées s'annullent, on obtient finalement un ensemble de 3 équations à 3 inconnues

$$w_0\, r_{xx}(0) + w_1\, r_{xx}(-1) + w_2\, r_{xx}(-2) = r_{xy}(0)$$

$$w_0\, r_{xx}(+1) + w_1\, r_{xx}(0) + w_2\, r_{xx}(-1) = r_{xy}(+1)$$

$$w_0\, r_{xx}(+2) + w_1\, r_{xx}(+1) + w_2\, r_{xx}(0) = r_{xy}(+2)$$

que l'on écrit sous la forme matricielle suivante

$$\begin{pmatrix} r_{xx}(0) & r_{xx}(-1) & r_{xx}(-2) \\ r_{xx}(+1) & r_{xx}(0) & r_{xx}(-1) \\ r_{xx}(+2) & r_{xx}(+1) & r_{xx}(0) \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} r_{xy}(0) \\ r_{xy}(1) \\ r_{xy}(2) \end{pmatrix} \tag{16.41}$$

Cette matrice d'autocorrélation est obligatoirement symétrique car la fonction d'autocorrélation est paire.

En représentant la matrice d'autocorrélation par le symbole R_{xx}, le vecteur des paramètres par W et le vecteur d'intercorrélation par r_{xy}, ce résultat s'écrit plus succinctement sous la forme :

$$R_{xx}\, W = r_{xy} \tag{16.42}$$

dont la solution est

$$W = R_{xx}^{-1}\, r_{xy} \tag{16.43}$$

Cette équation porte le nom d'équation normale ou de Wiener-Hopf.

16.3.3 Description matricielle

Les calculs que l'on vient d'exposer peuvent être présentés dans une écriture plus concise fréquemment utilisée. Définissant tout d'abord les vecteurs colonnes suivants :

$$W = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{p-1} \end{bmatrix} \qquad X(n) = \begin{bmatrix} x(n) \\ x(n-1) \\ \vdots \\ x(n-p+1) \end{bmatrix} \tag{16.44}$$

on obtient :

– le signal estimé $y_w(n)$

$$y_w(n) = \sum_{i=0}^{p-1} w_i\, x(n-i) = W^T\, X(n) = X(n)^T\, W \tag{16.45}$$

– l'erreur d'estimation $\varepsilon(n)$

$$\varepsilon(n) \;=\; y(n) - y_w(n) = y(n) - X(n)^T W \tag{16.46}$$

– l'erreur quadratique $\varepsilon^2(n)$

$$\varepsilon^2(n) \;=\; \Big(y(n) - X(n)^T\, W\Big)^2 \tag{16.47}$$

$$\varepsilon^2(n) \;= y^2(n) - 2\, y(n)\, X(n)^T\, W + W^T\, X(n)\, X(n)^T\, W \tag{16.48}$$

– l'erreur quadratique moyenne $J(W)$ fonction des paramètres W

$$\begin{aligned} J(W) \;&=\; E\{\varepsilon^2(n)\} = E\{\Big(y(n) - X(n)^T\, W\Big)^2\} \\ &=\; E\{y^2(n)\} - 2\, E\{y(n)\, X(n)^T\, W\} + E\{W^T\, X(n)\, X(n)^T\, W\} \end{aligned}$$

d'où

$$J(W) = r_{yy}(0) - 2\, r_{xy}^T\, W + W^T R_{xx} W \tag{16.49}$$

– le gradient de $J(W)$ par rapport au vecteur W des coefficients w_k

$$\frac{dJ}{dW} = -2\, r_{xy} + 2\, R_{xx}\, W \tag{16.50}$$

– le vecteur des paramètres optimaux qui annule le gradient

$$W = R_{xx}^{-1}\, r_{xy} \tag{16.51}$$

16.3.4 Applications du filtrage de Wiener

Les applications du filtrage de Wiener diffèrent par la manière dont est extraite la réponse désirée. Dans ce contexte, on peut distinguer quatre classes fondamentales utilisant le filtrage de Wiener :

1. l'identification de processus ; dans ce cas, on souhaite trouver la réponse impulsionnelle $w(n)$ représentant au mieux le processus inconnu ;
2. la modélisation inverse avec laquelle on tente de reconstruire un signal ;
3. la prédiction linéaire qui, sur la base des échantillons précédents, permet d'estimer une valeur à venir (codage LPC de la parole) ;
4. la suppression d'un signal perturbateur.

Dans ce qui suit, compte tenu du cadre dans lequel est présentée cette note, on se contentera d'illustrer comment on peut supprimer une perturbation grâce au filtrage adaptatif.

16.4 Suppression d'une perturbation

Comme illustration du filtrage de Wiener, imaginons la mesure de l'activité cardiaque d'un foetus à l'aide d'un électrocardiogramme (ECG) pris au niveau de l'abdomen de la mère et qui, naturellement, est perturbé par l'ECG de celle-ci.

Cette mesure nécessite l'utilisation de 2 capteurs. Avec le premier, on mesure le signal de référence $x(n)$ représentant, si possible, uniquement l'ECG de la mère. Avec le deuxième, on mesure le signal $y(n)$ qui est l'ECG du foetus perturbé par l'activité cardiaque de la mère.

Les signaux du schéma de Wiener (figure 16.4) sont alors les suivants :

1. $x(n)$ = l'ECG maternel mesuré près du coeur,
2. $y(n)$ = l'ECG foetal perturbé par celui de la mère,
3. $y_p(n)$ = l'ECG maternel près du foetus,
4. $y_w(n)$ = l'estimation de l'ECG maternel près du foetus,
5. $e(n)$ = l'ECG foetal,
6. $\varepsilon(n)$ = l'estimation de l'ECG foetal.

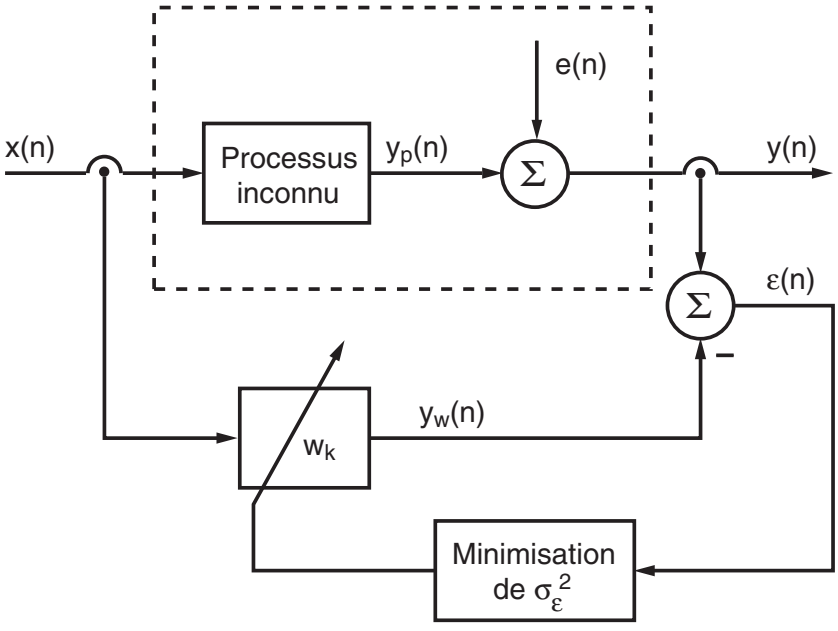

Fig. 16.4: Suppression de la perturbation $y_p(n)$

On notera que dans ce problème, les rôles sont inversés par rapport à la définition initiale du filtre de Wiener. En effet, le signal $e(n)$ considéré plus haut comme une perturbation du signal recherché $y_p(n)$, est, dans notre cas, le signal ECG que l'on souhaite mesurer et le signal $y_p(n)$ est la perturbation que l'on souhaite rejeter. C'est en recherchant $y_w(n) \simeq y_p(n)$ que l'on obtient une bonne estimation $\varepsilon(n)$ de l'ECG foetal $e(n) \simeq x(n) - y_w(n)$.

Dans cette simulation et dans un but didactique, on a choisi un modèle MA constitué de deux coefficients seulement $W = [w_0, w_1]^T$. Le système à résoudre s'écrit alors :

$$\begin{pmatrix} r_{xx}(0) & r_{xx}(1) \\ r_{xx}(1) & r_{xx}(0) \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \end{pmatrix} = \begin{pmatrix} r_{xy}(0) \\ r_{xy}(1) \end{pmatrix}$$

La recherche des coefficients W peut se faire de deux manières :

1. Dans le cas où l'on considère que le processus générateur de la perturbation est stationnaire, on commence par enregistrer la totalité des signaux $x(n)$ et $y(n)$. Puis, on calcule le vecteur des coefficients W après avoir calculé R_{xx} et r_{xy} pour l'ensemble des points acquis.
2. Si les signaux ne sont pas stationnaires (ce qui est le cas lorsque le processus change au cours du temps), il faut, après chaque échantillonnage, calculer les coefficients $W = R_{xx}^{-1}\, r_{xy}$.

16.4.1 Filtrage de Wiener classique

La figure 16.5 présente les résultats que l'on obtient avec un filtrage de Wiener classique dans lequel l'ensemble des points acquis est analysé en une seule fois. Dans cette approche, on fait l'hypothèse que le processus générateur de la perturbation est stationnaire. Sur la figure 16.5, on a tracé dans l'ordre :

– le signal $x(n)$ correspondant à l'ECG maternel,
– le signal $y(n)$ correspondant à l'ECG foetal perturbé,
– l'estimation $\varepsilon(n)$ de l'ECG foetal et sa valeur exacte $e(n)$ en pointillé.

On peut relever à quel point le résultat obtenu est proche du signal original. Un exemple de codage pour 2 paramètres est donné à la figure 16.6.

16.4.2 Remarque

D'un point de vue pratique, le filtre de Wiener tel qu'il a été présenté ci-dessus souffre de quelques limitations :

– il nécessite le calcul de la matrice d'autocorrélation R_{xx} et du vecteur d'intercorrélation r_{xy}, tous deux coûteux en temps de calcul ;
– il faut inverser la matrice R_{xx}, ce qui peut demander beaucoup de calcul et d'espace mémoire ;
– si les signaux ne sont pas stationnaires (ce qui est fréquent), R_{xx} et r_{xy} évoluent au cours du temps ; il faut donc à chaque instant résoudre l'équation de Wiener-Hopf.

Pour des applications en temps réel, il faut donc trouver un moyen rapide, efficace et robuste pour calculer récursivement la solution $W = R_{xx}^{-1} r_{xy}$. C'est ce que fait le filtrage adaptatif.

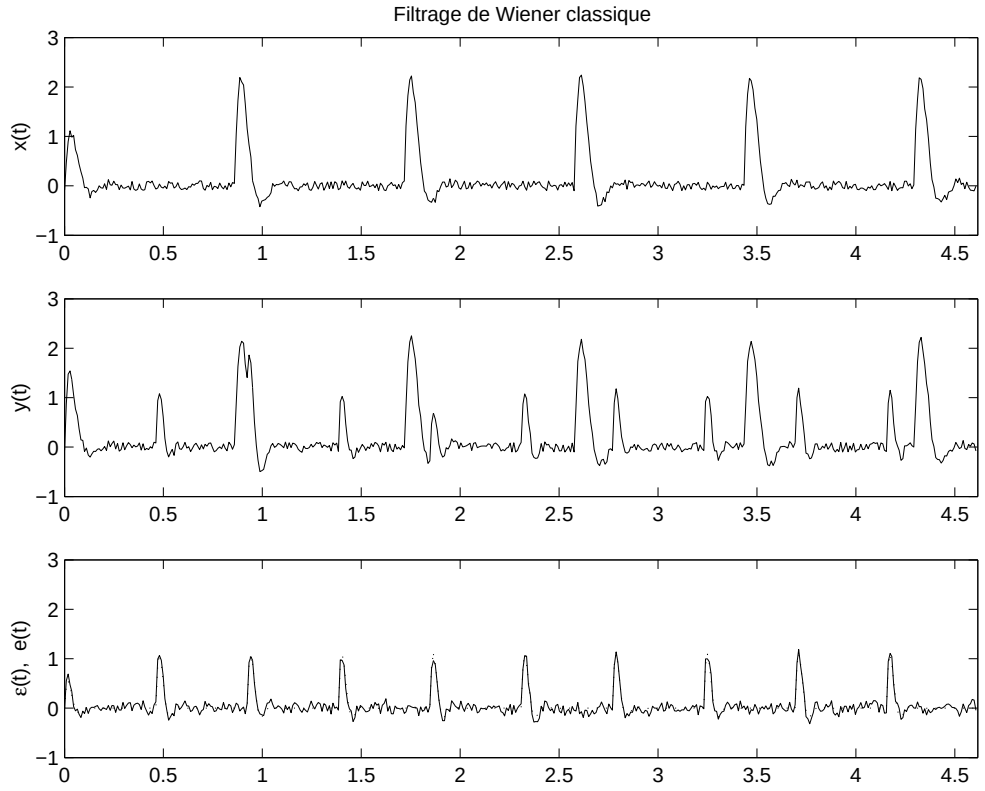

Fig. 16.5: Suppression d'une perturbation par filtrage de Wiener

16.5 Filtrage adaptatif

Un filtre adaptatif est un système numérique dont les coefficients se modifient eux mêmes en fonction des signaux extérieurs. Il est utilisé chaque fois qu'un environnement est mal connu ou changeant ou pour supprimer des perturbations situées dans le domaine de fréquences du signal utile, ce que les filtres classiques ne peuvent pas faire.

Un filtre adaptatif est constitué de deux parties distinctes :
– un filtre numérique à coefficients ajustables ;
– un algorithme de modification des coefficients basé sur un critère d'optimisation.

16.5.1 Algorithme récursif des moindres carrés (RLMS)

Nous avons vu au paragraphe 16.3.2 que pour trouver les paramètres optimaux, il faut descendre le long d'un paraboloïde afin atteindre le minimum de l'erreur quadratique moyenne. Mathématiquement, cette descente se fait dans le sens opposé à celui du gradient

$$\frac{\partial J}{\partial W} = -2\,r_{xy} + 2\,R_{xx}W$$

et on atteint le point optimum lorsque le gradient s'annule. La valeur des paramètres est alors donnée par la solution

$$W = R_{xx}^{-1}\,r_{xy}$$

```
% les signaux mesures sont stockes dans les vecteurs xt et yt
% initialisation des calculs
  rxx0 = 0 ; rxx1 = 0 ;
  rxy0 = 0 ; rxy1 = 0 ;
% boucle de calculs
  kmax = length(xt)-1 ;
  for n = 1 :kmax
    % lecture des signaux
    xn = xt(n) ;
    yn = yt(n) ;
    xn1 = xt(n+1) ;
    yn1 = yt(n+1) ;
    % correlation
    rxx0 = rxx0 + xn*xn ;
    rxx1 = rxx1 + xn*xn1 ;
    rxy0 = rxy0 + xn*yn ;
    rxy1 = rxy1 + xn*yn1 ;
  end ;
% solution de Wiener-Hopf
  Rxx = [rxx0 rxx1 ;
         rxx1 rxx0]
  rxy = [rxy0 ; rxy1]
  w = inv(Rxx) * rxy
% calcul du signal recherche
  xn1 = 0 ;
  for n = 0 :kmax-1
    xn = xt(n+1) ;
    yn = yt(n+1) ;
    ew(n+1) = yn - [xn, xn1]*w ;
    xn1 = xn ;
  end ;
```

FIG. 16.6: Exemple de codage d'un filtre de Wiener

De manière heuristique, on imagine bien que cette solution peut être atteinte récursivement en corrigeant les valeurs des coefficients w_k en chaque instant n dans le sens opposé à l'évolution de l'erreur quadratique par rapport au vecteur des coefficients $W(n)$ (figure 16.7) :

$$W(n) = W(n-1) - \frac{\gamma}{2}\left(\frac{\partial \varepsilon^2(n)}{\partial W}\right) \tag{16.52}$$

où γ est un facteur de pondération du gradient.

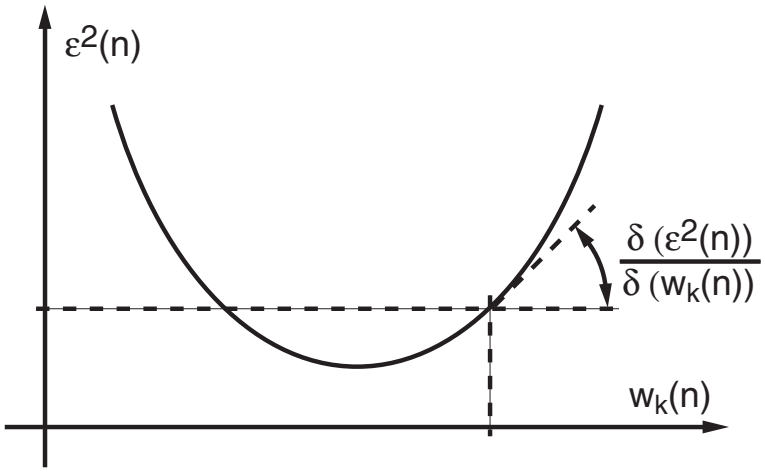

Fig. 16.7: Erreur quadratique $\varepsilon^2(n)$ en l'instant n et sa dérivée par rapport au coefficient $w_k(n)$

Comme l'erreur quadratique à l'instant n vaut :

$$\varepsilon^2(n) = \left(y(n) - \sum_{i=0}^{p-1} w_i\, x(n-i)\right)^2 = \left(y(n) - X(n)^T\, W\right)^2$$

il vient :

$$\frac{\partial \varepsilon^2(n)}{\partial W} = 2\,\varepsilon(n)\,\frac{\partial \varepsilon(n)}{\partial W} = -2\,\varepsilon(n)\,X(n)$$

On en déduit que la recherche de l'optimum peut se faire avec l'algorithme récursif suivant

$$W(n) = W(n-1) + \gamma\;\varepsilon(n)\,X(n) \tag{16.53}$$

que l'on désigne sous le nom d'algorithme RLMS (*Recursive Least Mean Square*).

Les grandeurs dont on a besoin sont :
– le vecteur des p coefficients à l'instant $n-1$:

$$W(n-1) = [w_0(n-1), w_1(n-1), \cdots, w_{p-1}(n-1)]^T$$

– les p dernières valeurs du signal d'entrée :

$$X(n) = [x(n),\, x(n-1),\, \cdots,\, x(n-p+1)]^T$$

– la valeur du signal de sortie $y(n)$ pour calculer l'écart à l'instant n

$$\varepsilon(n) = y(n) - \sum_{i=0}^{p-1} w_i \, x(n-i) \tag{16.54}$$

– le gain d'adaptation γ de l'algorithme récursif (généralement très inférieur à 1).
La valeur du gain d'adaptation γ est difficile à fixer : si on la choisit trop faible, la convergence vers la valeur optimum est très lente ; si on la choisit trop forte, la convergence se fait en oscillant onguement autour de la valeur optimum ; enfin, si le gain d'adaptation est trop élevé, le processus d'optimisation diverge.

Les avantages de cet algorithme résident dans la simplicité à le déduire, à le programmer, et au peu de calculs à effectuer. Par contre, ses inconvénients sont la lente convergence des paramètres et le risque d'oscillations ou de divergence si le gain d'adaptation est trop grand. Ces inconvénients, associés au fait que les signaux sont généralement non stationnaires, ont nécessité la recherche d'une adaptation automatique du gain.

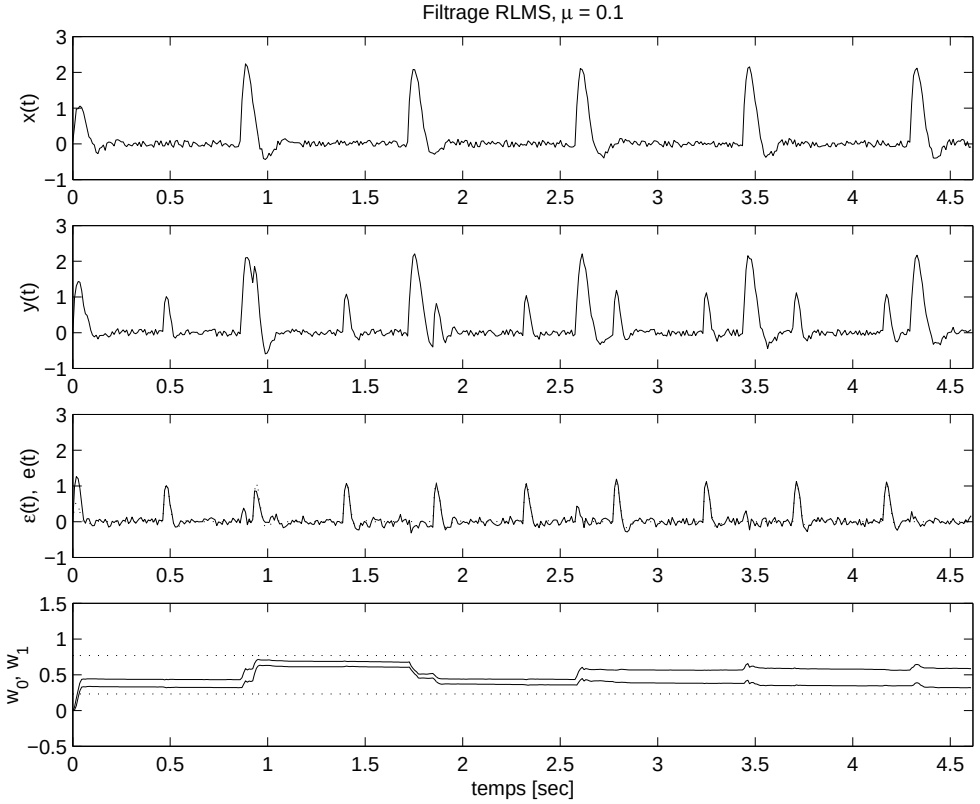

FIG. 16.8: Filtrage avec l'algorithme RLMS

16.5.2 Algorithme récursif normalisé (NLMS)

En 1975, Widrow et ses coauteurs ont montré qu'un gain d'adaptation stable compris entre 0 et 1 peut être utilisé si on le normalise par le nombre p de paramètres du vecteur W et par la puissance ou variance σ_x^2 du signal d'entrée $x(n)$.

Gain d'adaptation normalisé

Pour la plupart des situations pratiques, on choisit un gain initial $\gamma_0 \simeq 0.1$ qui, après normalisation par le nombre de paramètres et par la variance du signal d'entrée, donne un gain d'adaptation qui évolue en fonction de la puissance du signal d'entrée :

$$\gamma = \frac{\gamma_0}{p \cdot \sigma_x^2} \tag{16.55}$$

De manière à éviter que le gain n'augmente indéfiniment lorsque la puissance du signal de référence tend vers zéro, on peut corriger le dénominateur du gain en y ajoutant un terme constant $a \ll 1$:

$$\gamma = \frac{\gamma_0}{a + p \cdot \sigma_x^2} \tag{16.56}$$

L'algorithme s'écrit alors :

$$W(n) = W(n-1) + \frac{\gamma_0}{a + p \cdot \sigma_x^2}\, \varepsilon(n)\, X(n) \tag{16.57}$$

Comme cet algorithme utilise un gain normalisé par la puissance σ_x^2 du signal $x(n)$, il porte le nom d'algorithme NLMS (*Normalised Least Mean Square*).

Puissance moyenne du signal de référence

Dans le cas où le signal $x(n)$ n'est pas stationnaire, on doit évaluer sa puissance $P_x \equiv \sigma_x^2$ en tout instant :

$$P_x(n) = \frac{1}{n+1} \sum_{k=0}^{n} x^2(k)$$

Cette valeur moyenne peut également être évaluée à l'aide d'un filtre passe-bas en oubliant progressivement les anciennes valeurs.

Se souvenant qu'un filtre passe-bas d'ordre 1 et de gain unité, d'entrée $e(n)$ et de sortie $s(n)$ est décrit par sa fonction de transfert

$$H(z) = \frac{S(z)}{E(z)} = \frac{1-\lambda}{1-\lambda\, z^{-1}}, \qquad 0 < \lambda < 1 \tag{16.58}$$

ou, de manière équivalente, par son équation récursive

$$s(n) = (1-\lambda)\, e(n) + \lambda\, s(n-1) \tag{16.59}$$

le calcul de $P_x(n)$ se fait de la manière suivante :

$$P_x(n) = (1-\lambda)\, x^2(n) + \lambda\, P_x(n-1) \tag{16.60}$$

avec $\lambda = 0.90 \cdots 0.98$ suivant l'horizon de mémoire N désiré

$$N \simeq 3\, K_c = \frac{3}{|\ln(\lambda)|}$$

On montre aisément qu'au-delà de $N = 3/|\ln(\lambda)|$, la contribution des anciennes valeurs est inférieure à 5%. On peut relever que l'horizon de mémoire N vaut 30 pour $\lambda = 0.90$ ou 150 lorsque $\lambda = 0.98$.

Résultats

Les résultats ainsi obtenus sont présentés à la figure 16.9. Ils illustrent à l'évidence la rapidité de la convergence et la qualité des résultats qui est pratiquement aussi bonne que celle obtenue avec le filtrage optimum de Wiener.

La figure 16.10 montre comment le gain change au cours du temps ; on y voit nettement son augmentation lorsque la puissance du signal de référence est faible. Une partie du codage est présenté dans la figure 16.11.

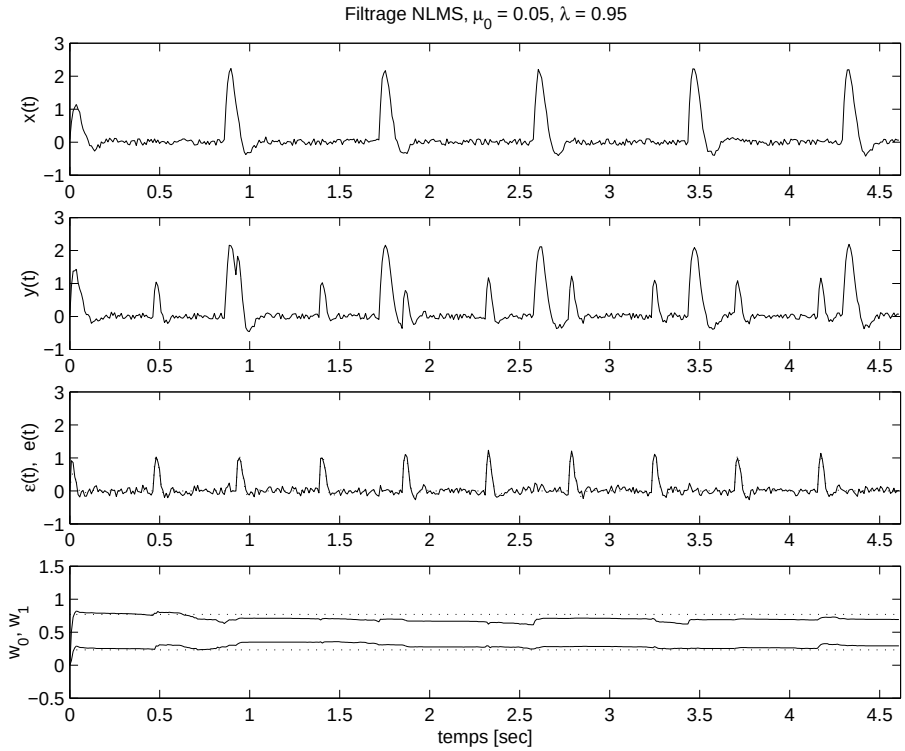

FIG. 16.9: Filtrage avec l'algorithme NLMS

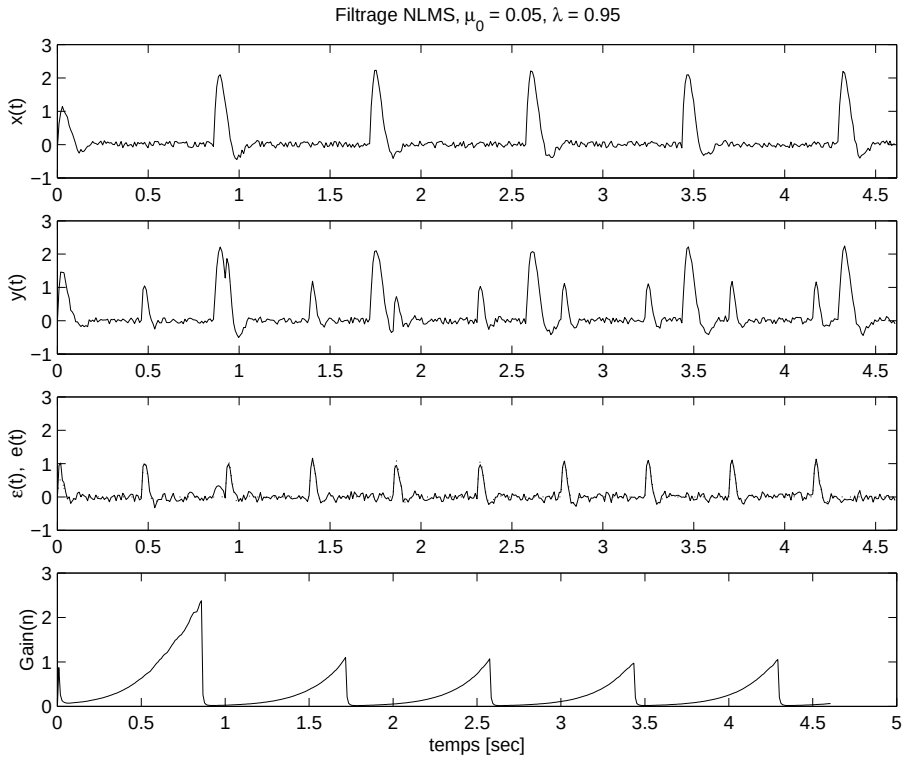

Fig. 16.10: Évolution du gain de l'algorithme NLMS

```
% constantes
  p = 3 ; % modifier selon les besoins
  gamma0 = 0.1 ;
  lambda = 0.95 ;
% initialisation des calculs
  Wn = zeros(p,1) ;
  Pxx = 0 ;
  a = 1e-3 ;
% boucle de calculs
for n = p-1 :kmax-1
  % signaux a l'instant n
  xn = xt(n+1) ;
  yn = yt(n+1) ;
  % puissance moyenne de x(n)
  Pxx = (1-lambda)*xn^2 + lambda*Pxx ;
  % calcul des paramètres W(n)
  Xn = xt(n+1 :-1 :n-p+2) ;
  en = yn - Xn'*Wn ;
  Wn = Wn + gamma0/(a+p*Pxx) * en*Xn ;
  % memorisation du signal recherche et des parametres
  ew(n+1) = en ;
  wt(n+1, :) = Wn' ;
end ;
```

FIG. 16.11: Exemple de codage d'un filtre NLMS

16.6 Exercices

RL 1

Considérant l'ensemble des notes n suivantes :

2.8	4.7	3.2	4.2	2.6	4.8	3.5	5.4	5.4	4.4
4.0	5.6	5.3	4.6	5.3	4.6	3.4	3.2	3.4	4.2

1. Calculez la note moyenne et sa déviation standard.
2. Comparez aux résultats obtenus avec les fonctions Matlab `mean`, `var` et `std`.
3. Calculez puis tracez à la main la répartition des notes telles que

$$n < 3, \quad 3 \leq n < 4, \quad 4 \leq n < 5, \quad 5 \leq n \leq 6$$

4. Commentez vos résultats.

RL 2

La mesure de la caractéristique statique d'un amplificateur a donné les résultats suivants :

U_{in} [mV]	-50	-40	-30	-20	-10	0	10	20	30	40	50
U_{out} [V]	-3.69	-3.38	-2.43	-1.68	-0.56	0.17	0.87	1.96	2.45	3.28	3.76

1. Tracez cette caractéristique ; qu'en pensez-vous ?
2. Admettant que l'amplificateur est linéaire, utilisez la régression linéaire (équations de la section 16.2.3) pour calculer à la main son gain et sa tension de décalage. Puis :
   a) Comparez aux résultats fournis par `polyfit`.
   b) Tracez sur un même graphe les points mesurés et la caractéristique linéaire de l'amplificateur avec une abscisse fine ($\Delta U_{in} = 1\,mV$).
   c) Calculez la puissance des écarts entre le modèle et la mesure.
3. Considérant que l'amplificateur n'est pas parfaitement linéaire, utilisez la fonction `polyfit` pour calculer le polynôme d'ordre 3 (pourquoi 3 et pas 2 ou 4 ?) passant au mieux parmi ces points.
   a) Y a-t-il un sens à donner le gain de l'amplificateur et sa tension de décalage ?
   b) Tracez sur un même graphe les points mesurés et la caractéristique non linéaire de l'amplificateur avec une abscisse fine ($\Delta U_{in} = 1\,mV$).
   c) Calculez la puissance des écarts entre le modèle et la mesure ?
4. Répétez le point 3 avec un polynôme d'ordre 9. Commentez la puissance des écarts et l'allure de ce polynôme par rapport aux modèles précédents.

Prb 1

On s'intéresse ici aux notions de base des statistiques et probabilités. Pour cela :

1. Générez les trois signaux suivants avec $N = 10'000$ et $n = 0 : N-1$,

$$x_u(n) = \text{rand}(size(n)), \quad x_g(n) = 0.5\,\text{randn}(size(n)), \quad x_s(n) = \sin(2\pi\, n\,/N)$$

2. Tracez ces signaux par rapport à n (`subplot(3,1,k)`) ; observez-les avec le zoom ; commentez vos observations.
3. Quels résultats fournissent les fonctions `rand` et `randn` ?
4. Pouvez-vous donner une estimation des valeurs moyennes et variances des trois signaux ?
5. Pour chacun des 3 signaux, calculez leurs valeur moyenne, variance et écart-type ; vérifiez l'équation (16.3) ; commentez.
6. Tracez et comparez les 3 histogrammes (`hist(xn,round(sqrt(N)))`).
7. Commentez ces résultats d'un point de vue statistique ; en particulier, que pensez-vous de la probabilité d'apparition de certaines valeurs ?

Prb 2

Avec l'exercice précédent vous avez compris que l'histogramme permet de compter le nombre de fois n_k où une valeur x se situe dans la case k de largeur

$$\Delta x = \frac{x_{max} - x_{min}}{N_k}, \quad \text{avec} \quad N_k = \text{nombre de cases}$$

La probabilité de se trouver dans une case est donc égale au contenu de la case divisé par le nombre total de points N. On définit ainsi la fonction de probabilité discrète

$$p(k) \equiv p(k\Delta x \leq x < (k+1)\Delta x) = \frac{n_k}{N} \tag{16.61}$$

On peut également considérer la somme cumulée des valeurs de l'histogramme et normaliser son maximum à 1. On obtient alors la fonction de répartition des probabilités définie comme la probabilité que la valeur de x soit inférieure à une valeur donnée $X = K\,\Delta x$. Mathématiquement, cela s'écrit :

$$P(K) \equiv p(-\infty < x \leq K\,\Delta x) = \sum_{k\to-\infty}^{K} p(k) \tag{16.62}$$

Cette approche permet de calculer simplement la probabilité que la valeur de x se situe dans un domaine donné ; on a en effet

$$P(K_1\Delta X < x \leq K_2\Delta X) = \sum_{k=K_1+1}^{K_2} p(k) = P(K_2) - P(K_1) \tag{16.63}$$

Dans le cas d'une variable continue, la fonction de probabilité devient une densité de probabilité définie comme suit

$$p(x) = \lim_{N\to\infty,\,\Delta x\to 0} \left(\frac{n_k}{N\,\Delta x}\right) \tag{16.64}$$

et la fonction de répartition des probabilités s'écrit alors

$$P(x : x' < x) = \int_{-\infty}^{x} p(x)\, dx \tag{16.65}$$

Afin de bien comprendre ce qui précède, je vous propose d'appliquer les points ci-après sur les 3 signaux suivants
- un signal aléatoire $x_u[n]$ à distribution uniforme compris entre -4 et $+4$;
- un signal aléatoire $x_g[n]$ à distribution gaussienne de variance unité ;
- un signal sinusoïdal $x_s[n]$ d'amplitude 4 et de période N ;

où $N = 10'000$ et $n = 0 : N - 1$.

1. Représentez le signal avec
   `subplot(3,1,1) ; plot(nn,xn)`.
2. Calculez sa valeur moyenne et sa variance ; comparez avec les valeurs théoriques.
3. Calculez son histogramme avec
   `[nk,xk] = hist(xn,sqrt(N))`.
4. Calculez et représentez la fonction de probabilité $p(k)$ avec
   `subplot(3,1,2) ; plot(xk,pk)`.
5. Calculez et représentez la répartition des probabilités $P(K)$ avec
   `PK = cumsum(pk) ; subplot(3,1,3) ; plot(xk,PK)`.
6. Calculez la probabilité de trouver une valeur de x comprise entre -1 et $+1$ (la fonction `find` facilite ce calcul).
7. Analysez et commentez vos graphes et résultats.

Prb 3

Appliquez ce que vous venez de voir à des signaux réels (sons et images). Plus précisément,

1. Chargez le fichier sons : `xt = load('colibri.txt') ;` normalisez sa valeur maximum à 1 : `xt = xt/max(abs(xt)) ;`.
2. Tracez son graphe et écoutez-le avec `wavplay(xt,8000) ;`.
3. Calculez sa valeur moyenne, sa puissance et sa variance.
4. Tracez son histogramme. Quel est le modèle de distribution qui vous paraît approprié pour décrire le signal ?
5. Ajustez les modèles choisis sur la courbe de probabilité réelle. Commentez.
6. Chargez le fichier image : `img = imread('lena256.jpg') ;` visualisez son contenu et transformez la matrice image en un vecteur : `img_vecteur = double(img( :)) ;`
7. Répétez les points 1 à 5 ci-dessus.

Corr 1

Dans le but de vous familiariser avec les résultats de la corrélation, appliquez la fonction Matlab de corrélation `rxy = xcorr(y,x,L,'unbiased')` à chacun des 3 signaux suivants

$$x_c[n] = \sin(2\pi\, n\,/N), \quad x_q[n] = \text{square}(2\pi\, n\,/N), \quad x_g[n] = \text{randn}(size(n))$$

où $n = 0 : 1000$ et $N = 50$. Pour ce faire :

1. calculez puis tracez les 3 signaux dans une fenêtre ;
2. calculez puis tracez les 3 fonctions d'autocorrélation dans une nouvelle fenêtre ;
3. quelle est l'utilité du paramètre L ?
4. vous souvenant que $r_{xx}[0] = P_x$, que doit valoir le maximum de chaque autocorrélation ?
5. calculez et tracez l'intercorrélation entre $x_c[n]$ et $x_q[n]$ (prenez garde à l'ordre des signaux `rcq = xcorr(xq,xc,L,'unbiased')`).

WH 1

Considérant un filtre MA causal décrit par ses coefficients $w[0 \cdots 2] = [3, 2, 1]$,

1. dessinez le schéma fonctionnel de ce filtre en prenant $x[n]$ et $y[n]$ comme signaux d'entrée et de sortie ;
2. montrez que, si on lui applique le signal $x(n) = [+1, -1, +1, -1, +1]$, la sortie vaudra $y[n] = [+3, -1, +2, -2, +2]$;
3. pour faire ce calcul, quelle modification implicite avez-vous appliqué à $x[n]$?
4. calculez $y[n]$ avec Matlab.

WH 2

Ayant appliqué le signal $x[n] = [+1, +1, -1, +1, -1, 0]$ à un système MA décrit par l'équation

$$y[n] = w_0 x[n] + w_1 x[n-1]$$

on a obtenu en sortie le signal $y[n] = [+2, +3, -1, +1, -1, -1]$. Utilisez le filtrage de Wiener pour trouver les paramètres w_0, w_1 sans Matlab. Pour ce faire :

1. Calculez à la main les fonctions de corrélation $r_{xx}([k]$ et $r_{xy}[k]$ pour k compris entre –5 et +5.
2. Quelles sont les valeurs de corrélation dont vous avez besoin pour résoudre ce problème ?
3. Écrivez les vecteurs et matrices r_{xx}, r_{xy}, R_{xx} nécessaires pour le filtrage de Wiener.
4. Résolvez le système $R_{xx}\, W = r_{xy}$.
5. A-t-on besoin de 5 couples de valeurs (x, y) pour trouver w_0 et w_1 ? Justifiez et vérifiez votre réponse.
6. Quel est l'avantage d'avoir un grand nombre de valeurs (x, y) ?

WH 3

Résolvez le problème précédent avec l'aide de Matlab. Pour ce faire, utilisez

1. la fonction `rxy = xcorr(y,x,p-1)` où p est le nombre de paramètres (prenez garde à l'ordre des vecteurs `x` et `y`) ;
2. la fonction `Rxx = toeplitz(rxx)` pour remplir la matrice de corrélation R_{xx}.

WH 4

Dans ce qui suit on souhaite extraire un signal inconnu $s(n)$ fortement perturbé par le réseau électrique de fréquence 50 Hz. Pour ce faire, on a mesuré simultanément le signal du réseau $x(n)$ et le signal bruité $y(n)$ en les échantillonnant à la fréquence f_e de 10 kHz. Pour résoudre ce problème,

1. chargez les signaux contenus dans le fichier `xy50hz.txt` :
```
signaux = load('xy50hz.txt') ;
xn = signaux( :,1) ;
yn = signaux( :,2) ;
N = length(xn) ;
nn = 0 :Te :(N-1)*Te ;
```
2. tracez $x(n)$ et $y(n)$;
3. dessinez le schéma de Wiener ; où se trouve le signal $s[n]$?
4. recherchez $s[n]$ en appliquant l'algorithme de Wiener-Hopf avec 2 paramètres ;
5. augmentez le nombre de paramètres p ; observez leurs valeurs et la puissance de $s[n]$; concluez ;
6. calculez le rapport signal sur bruit S_{eff}/Y_{eff}.

LMS 1

Il est possible d'évaluer en temps réel la puissance moyenne d'un signal en "oubliant" progressivement les valeurs anciennes. Ceci peut se faire de la manière suivante :

$$P_x[n] = (1-\lambda)\, x^2[n] + \lambda\, P_x[n-1]$$

1. Montrez que cet algorithme est l'équivalent d'un filtre passe-bas d'ordre 1. Pour cela :
   a) dessinez son schéma fonctionnel ;
   b) calculez sa fonction de transfert ;
   c) que valent l'instant caractéristique et l'horizon de mémoire à 5% ?
2. Imaginez (sans l'aide de Matlab !) un signal $x(t)$ composé d'une sinusoïde d'amplitude $A_1 = 1\,V$, de fréquence $f_0 = 1000\,Hz$ et de durée $t_{max} = 0.1\,sec$ suivi de la même sinusoïde d'amplitude $A_2 = 2\,V$. Comment évolue la puissance de ce signal ? Que vaut-elle ?
3. Générez le signal $x[n]$; quelle fréquence d'échantillonnage prenez-vous ?
4. Calculez sa puissance moyenne avec l'algorithme ci-dessus en prenant $\lambda = 0.95$ et $\lambda = 0.99$; quel est l'horizon de mémoire à 5% correspondant à ces deux valeurs ?
5. Tracez $P_x[n]$; concluez.

LMS 2

Dans ce problème, on souhaite diminuer le bruit environnant lors d'une conversation téléphonique en utilisant un deuxième microphone placé sur le côté extérieur du téléphone. Le microphone de base capte le message entaché du bruit environnant, alors que le deuxième capte seulement le bruit. Grâce à l'algorithme LMS, il est possible d'améliorer sensiblement la qualité du message.

Pour le vérifier :

1. Dessinez le schéma de Wiener correspondant à ce problème et précisez quels sont les signaux en présence.
2. Le fichier `bjrbruit.dat` constitué de trois colonnes contient trois signaux enregistrés à la fréquence $f_e = 8\,kHz$, à savoir, le bruit (microphone extérieur $= x[n]$), le message perturbé par le bruit (microphone-bouche $= y[n]$) et, dans un but de comparaison, le message non bruité. De ce fichier, extrayez les deux premiers signaux ; tracez-les et écoutez-les avec la fonction `soundsc(signal,fe)`.
3. Appliquez l'algorithme NLMS sur les deux premiers signaux pour diverses longueurs p du vecteur W.
4. Pour $p = 5$ et $p = 20$ par exemple, tracez les valeurs asymptotiques des composantes de $W(n \to \infty)$. Qu'en pensez-vous ?
5. Tracez le signal fourni par NLMS et l'évolution des paramètres $W[n]$.
6. Observez les spectrogrammes des trois signaux (`specgram(signal,128,fe)`) ; commentez vos observations.
7. Écoutez le signal extrait ; qu'en pensez-vous ?
8. Le temps de calcul sur un PC 500 MHz est prohibitif (environ quinze fois la durée du message). Sachant qu'en un cycle d'horloge, un DSP réalise une multiplication et une addition, pensez-vous qu'il soit possible de faire ces calculs en temps réel avec un DSP dont le temps de cycle est de 20 ns ? Justifiez votre réponse.

Bibliographie

[1] B. Widrow, S.D. Stearns : *Adaptive Signal Processing Algorithms*, Prentice Hall, 1985.

[2] S.D. Stearns, R.A. David : *Signal Processing Algorithms in Matlab*, Prentice Hall, 1996.

[3] E.C. Ifeachor, Q.W. Jervis : *Digital Signal Processing, A Practical Approach*, Addison Wesley, 1993.

[4] S. Haykin : *Adaptive Filter Theory*, Prentice Hall, 1991.