BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

ISO 9001:2008

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: ĐIỆN TỬ VIỄN THÔNG

Người hướng dẫn: Thạc sỹ Nguyễn Văn Dương
Sinh viên : Phan Thùy Ninh

HẢI PHÒNG - 2010

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

NGHIÊN CỨU BỘ LỌC THÍCH NGHI

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC CHÍNH QUY
NGÀNH : ĐIỆN TỬ VIỄN THÔNG

Người hướng dẫn : Thạc sỹ Nguyễn Văn Dương
Sinh viên : Phan Thùy Ninh

HẢI PHÒNG - 2010

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên : Phan Thùy Ninh . Mã số : 100218.

Lớp : ĐT1001. Ngành: Điện tử viễn thông.

Tên đề tài : Nghiên cứu bộ lọc thích nghi.

NHIỆM VỤ ĐỀ TÀI

1. Nội dung và các yêu cầu cần giải quyết trong nhiệm vụ đề tài tốt nghiệp (về lý luận, thực tiễn, các số liệu cần tính toán và các bản vẽ).

……..
……..
……..
……..
……..
……..
……..
……..
……..
……..

2. Các số liệu cần thiết để thiết kế, tính toán.

……..
……..
……..
……..
……..
……..
……..
……..
……..
……..
……..

3. Địa điểm thực tập tốt nghiệp.

……..
……..

……..

CÁN BỘ HƯỚNG DẪN ĐỀ TÀI TỐT NGHIỆP

Người hướng dẫn thứ nhất:

Họ và tên : Nguyễn Văn Dương

Học hàm, học vị: Thạc sỹ.

Cơ quan công tác : Trường Đại học Dân lập Hải Phòng.

Nội dung hướng dẫn

:.....

…….....……

……..

…….....

……..

…….....…

……..

…….....…

……..

Người hướng dẫn thứ hai:

Họ và tên

:.....

Học hàm, học vị

:.....

Cơ quan công tác

:.....

Nội dung hướng dẫn

:.....

…….....…

……..

…….....……
……..
…….....……
……..

Đề tài tốt nghiệp được giao ngày ..... tháng ..... năm 2010.

Yêu cầu phải hoàn thành xong trước ngày ..... tháng ..... năm 2010.

Đã nhận nhiệm vụ ĐTTN	Đã giao nhiệm vụ ĐTTN
Sinh viên	*Người hướng dẫn*

Hải Phòng, ngày ..... tháng ..... năm 2010.

HIỆU TRƯỞNG

GS.TS.NGƯT Trần Hữu Nghị

PHẦN NHẬN XÉT TÓM TẮT CỦA CÁN BỘ HƯỚNG DẪN

1. Tinh thần thái độ của sinh viên trong quá trình làm đề tài tốt nghiệp:

……..
……..
……..
……..
……..
……..
……..
……..
……..
……..
……..

2. Đánh giá chất lượng của đồ án (so với nội dung yêu cầu đã đề ra trong nhiệm vụ Đ.T.T.N trên các mặt lý luận, thực tiễn, tính toán số liệu...):

……..
……..
……..
……..
……..
……..
……..
……..
……..

3. Cho điểm của cán bộ hướng dẫn (ghi cả số và chữ) :

……..
……..
……..

Hải Phòng, ngày ..... tháng ..... năm 2010.

Cán bộ hướng dẫn

PHẦN NHẬN XÉT TÓM TẮT CỦA NGƯỜI CHẤM PHẢN BIỆN

1. Đánh giá chất lượng đề tài tốt nghiệp về các mặt thu thập và phân tích số liệu ban đầu, cơ sở lý luận chọn phương án tối ưu, cách tính toán chất lượng thuyết minh và bản vẽ, giá trị lý luận và thực tiễn đề tài.

……..
……..
……..
……..
……..

……..
……..
……..
……..
……..

2. Cho điểm của cán bộ phản biện. (Điểm ghi cả số và chữ).

……..
……..
……..
……..
……..
……..
……..
……..
……..

Hải Phòng, ngày ..... tháng ..... năm 2010.

Người chấm phản biện

MỤC LỤC

LỜI NÓI ĐẦU

Sống trong thế giới hiện đại như ngày nay, chúng ta tiếp xúc với rất nhiều loại tín hiệu và dưới nhiều dạng khác nhau. Có các tín hiệu rất cần thiết như âm thanh, hình ảnh hay các tín hiệu giải trí như âm nhạc .v.v. Bên cạnh cũng luôn tồn tại các tín hiệu khó chịu hoặc không cần thiết trong hoàn cảnh riêng nào đó, mà ta gọi đó là nhiễu. Xử lý tín hiệu là trích lấy, tăng cường, lưu trữ và truyền thông tin có ích mà con người cần quan tâm trong vô vàn thông tin có ích cũng như vô ích đồng thời phải loại bỏ nhiễu, để từ đó có được thông tin mà không mất đi tính trung thực của thông tin gốc. Trong các hướng đi và các cách giải quyết khác nhau cho vấn đề nêu trên, thì lĩnh vực xử lý tín hiệu số(DSP) mỗi ngày càng phát triển mạnh mẽ và vững vàng. Trong đó không thể không nhắc tới vai trò của các bộ lọc, nhất là các bộ lọc nhiễu. Trong đồ án này, em thực hiện nghiên cứu về bộ lọc thích nghi, một loại lọc nhiễu được ứng dụng trong rất nhiều hệ thống thực tế. Đây là loại bộ lọc có thuật toán thay đổi để thích ứng được với tín hiệu vào. Đồ án gồm 3 chương:

Chương 1: Giới thiệu về bộ lọc số.

Chương 2: Nội dung nghiên cứu bộ lọc thích nghi.

Chương 3: Mô phỏng ứng dụng bộ lọc thích nghi.

Em xin cảm ơn thày Nguyễn Văn Dương, giảng viên hướng dẫn, đã rất nhiệt tình chỉ bảo để em hoàn thành đề tài nghiên cứu này, cũng như các thày cô khác trong bộ môn đã tạo điều kiện cho em trong suốt thời gian làm đề tài.

Hải Phòng, ngày 12 tháng 07 năm 2010

Sinh viên

Ninh

Phan Thùy Ninh

Chương 1.

BỘ LỌC SỐ

Bộ lọc số là hệ thống tuyến tính bất biến theo thời gian. Thông số vào và ra của hệ thống quan hệ với nhau bằng tổng chập

$$Y(Z)=H(Z).X(Z) \quad (1.1.1)$$

Chuyển đổi miền Z của đáp ứng xung đơn vị H(Z) được gọi là hàm hệ thống. Biến đổi Fourier của đáp ứng xung đơn vị $H(e^{j\omega})$ là một hàm phức của ω, biểu diễn theo phần thực và phần ảo là

$$H(e^{j\omega})=Hr(e^{j\omega})+jHi(e^{j\omega}) \quad (1.1.2)$$

Hoặc biểu diễn dưới dạng góc pha:

$$H(e^{j\omega}) = |H(e^{j\omega})| e^{j\arg|H(e^{j\omega})|} \quad (1.1.3)$$

Một hệ thống tuyến tính bất biến nhân quả là dạng có h(n)=0 với n<0. Một hệ thống ổn định là dạng với tất cả các thông số đưa vào hữu hạn sẽ có thông số ra hữu hạn.

Điều kiện cần và đủ cho một hệ thống tuyến tính bất biến ổn định là:

$$\sum_{n=-\infty}^{\infty} |h(n)| < \infty \quad (1.1.4)$$

Thêm vào đó, tất cả các hệ thống tuyến tính bất biến có các thông số vào và ra như các bộ lọc thoả mãn phương trình sai phân có dạng:

$$y(n) - \sum_{k=1}^{N} a_k y(n-k) = \sum_{r=0}^{M} b_r x(n-r) \quad (1.1.5)$$

Chuyển đổi sang miền Z cả hai vế của phương trình ta được:

$$H(Z) = \frac{Y(Z)}{X(Z)} = \frac{\sum_{r=0}^{M} b_r Z^{-r}}{1 - \sum_{k=1}^{N} a_k Z^{-k}} \quad (1.1.6)$$

So sánh hai phương trình trên, từ phương trình sai phân (1.1.3) ta có thể đạt được H(Z) trực tiếp bằng cách đồng nhất các hệ số của phần tử vào trễ trong (1.1.5) với các luỹ thừa tương ứng Z^{-1}.

Hàm hệ thống H(Z) là một hàm hữu tỉ của Z^{-1}. Nó có thể được biểu diễn bằng dạng điểm cực và điểm không trong mặt phẳng Z. Như vậy H(Z) có thể viết dạng:

$$H(Z) = \frac{A\prod_{r=1}^{M}(1 - c_r Z^{-1})}{\prod_{k=1}^{N}(1 - d_k Z^{-1})} \quad (1.1.7)$$

Như chúng ta đã xét trong miền Z, hệ thống nhân quả sẽ có miền hội tụ dạng $|Z| < R_1$. Nếu hệ thống cũng là ổn định thì R_1 phải nhỏ hơn giá trị đơn vị, do đó miền hội tụ bao gồm là vòng tròn đơn vị. Như vậy trong hệ thống bất biến, nhân quả thì tất cả các điểm cực của H(Z) phải nằm trong vòng tròn đơn vị. Để thuận tiện, ta phân thành các lớp hệ thống, những lớp này bao gồm hệ thống đáp ứng xung hữu hạn (Finit duration Impulse Response_FIR), và hệ thống đáp ứng xung vô hạn (Infinit duration Impulse Response_IIR).

1.1. Hệ thống FIR

Phương trình sai phân sẽ là:

$$y(n) = \sum_{r=0}^{M} b_r x(n-r) \quad (1.1.8)$$

chúng ta thấy rằng:

$$h(n) = \begin{cases} b_n & 0 \le n \le M \\ 0 & với\ các\ n\ còn\ lại \end{cases} \quad (1.1.9)$$

Hệ thống FIR có rất nhiều thuộc tính quan trọng, trước tiên chúng ta chú ý rằng H(Z) chỉ có điểm không là một đa thức của Z^{-1} và tất cả các điểm cực của H(Z) đều bằng không, tức là H(Z) chỉ có điểm không. Thêm nữa, hệ thống FIR có thể có chính xác pha tuyến tính. Nếu h(n) xác định theo công thức sau

$$h(n) = \pm h(M-n) \quad (1.1.10)$$

thì $H(e^{j\omega})$ có dạng

$$H(e^{j\omega}) = A(e^{j\omega}).e^{-j\omega M/Z} \quad (1.1.11)$$

$H(e^{j\omega})$ chỉ có phần thực hoặc phần ảo tuỳ thuộc vào phương trình (1.1.10) lấy dấu (+) hay dấu (-).

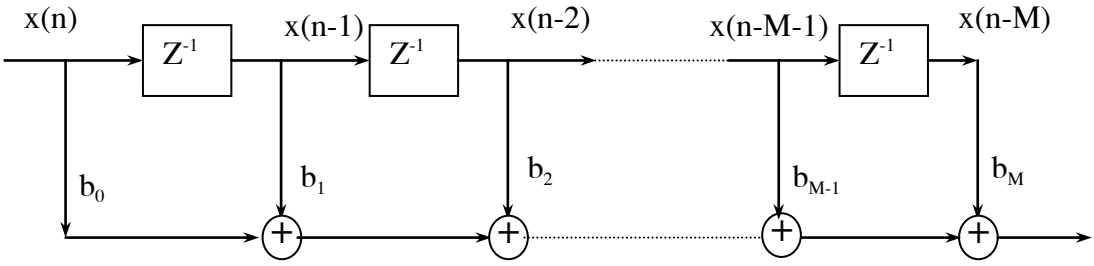
Dạng pha tuyến tính chính xác thường rất hữu ích trong các ứng dụng xử lý âm thanh, khi mà xác định thứ tự thời gian là cần thiết. Các thuộc tính này của bộ lọc FIR cũng có thể đơn giản hoá vấn đề xấp xỉ, nó chỉ xét đến khi đáp ứng độ lớn cần thiết. Khoảng sai số mà được bù để thiết kế các bộ lọc với đáp ứng xung pha tuyến tính chính xác là phần mà một khoảng thời gian tồn tại

đáp ứng xung phù hợp được yêu cầu để xấp xỉ phần nhọn bộ lọc bị cắt đi.

Dựa trên những thuộc tính chung với bộ lọc FIR pha tuyến tính, người ta đã phát triển ba phương pháp thiết kế xấp xỉ. Những phương pháp này là:

- Thiết kế cửa sổ
- Thiết kế mẫu tần số
- Thiết kế tối ưu

Chỉ có phương pháp đầu tiên là phương pháp phân tích, thiết kế khối khép kín tạo bởi các phương trình có thể giải để nhận được các hệ số bộ lọc. Phương pháp thứ hai và phương pháp thứ ba là phương pháp tối ưu hoá, nó sử dụng phương pháp lặp liên tiếp để được thiết kế bộ lọc



Hình 1.1. Mạng số cho hệ thống FIR

Bộ lọc số thường được biểu diễn dạng biểu đồ khối, như hình (1.1) ta biểu diễn phương trình sai phân (1.1.8). Sơ đồ như vậy thường được gọi là một cấu trúc bộ lọc số. Trên sơ đồ, biểu diễn các toán tử yêu cầu tính giá trị mỗi dãy ra từ giá trị của dãy đưa vào. Những phần tử cơ bản của sơ đồ biểu diễn ý nghĩa phép cộng, nhân các giá trị của dãy với hằng số (các hằng số trên nhánh hàm ý phép nhân), và chứa các giá trị trước của dãy vào. Vì vậy biểu đồ khối đưa ra chỉ dẫn rõ ràng về tính phức tạp của hệ thống.

1.2. Hệ thống IIR

Nếu hàm hệ thống của phương trình (1.1.7) có các điểm cực cũng như điểm không, thì phương trình sai phân (1.1.5) có thể viết:

$$y(n) = \sum_{k=1}^{N} a_k y(n-k) + \sum_{r=0}^{M} b_r x(n-r) \qquad (1.1.12)$$

Phương trình này là công thức truy hồi, nó có thể được sử dụng để tính giá trị của dãy ra từ các giá trị trước đó của thông số ra và giá trị hiện tại, trước đó của dãy đầu vào. Nếu M<N trong phương trình (1.1.7), thì H(Z) có thể biến đổi về dạng:

$$H(Z) = \sum_{k=1}^{N} \frac{A_k}{1 - d_k Z^{-1}} \tag{1.1.13}$$

Cho hệ thống nhân quả, ta dễ dàng biểu diễn

$$h(n) = \sum_{k=1}^{N} A_k (d_k)^n u(n) \tag{1.1.14}$$

Ta có thể thấy rằng dãy h(n) có chiều dài vô hạn. Tuy nhiên, vì công thức truy hồi (1.1.12) thường dùng để thực hiện bộ lọc IIR, nó sử dụng ít phép tính hơn là đối với bộ lọc FIR. Điều này đặc biệt đúng cho các bộ lọc lựa chọn tần số cắt nhọn.

Có nhiều phương pháp thiết kế sẵn có cho bộ lọc IIR. Những phương pháp thiết cho bộ lọc lựa chọn tần số (thông thấp, thông dải, ...) một cách chung nhất là dựa trên những biến đổi của thiết kế tương tự.

- Các thiết kế Butterword
- Các thiết kế Bessel
- Các thiết kế Chebyshev
- Các thiết kế Elliptic

Tất cả những phương pháp trên dùng phép phân tích tự nhiên và được ứng dụng rộng rãi để thiết kế các bộ lọc IIR. Thêm vào đó các phương pháp tối ưu hoá IIR đã được phát triển cho thiết kế xấp xỉ liệt kê, điều này không dễ thích nghi với một trong các phương pháp xấp xỉ trên.

Sự khác nhau chính giữa FIR và IIR là IIR không thể thiết kế để có pha tuyến tính chính xác, khi mà FIR có những thuộc tính này, còn bộ lọc IIR hiệu quả hơn trong thực hiện lọc cắt nhọn hơn là FIR.

Mạng bao hàm phương trình (1.1.12) được biểu diễn trong hình 1.2a cho trường hợp N=M=3, nó thường được gọi là dạng biểu diễn trực tiếp.

Đặc biệt bộ phương trình sau thường được sử dụng:

$$\begin{aligned} w(n) &= \sum_{k=1}^{N} a_k w(n-k) + x(n) \\ y(n) &= \sum_{r=0}^{M} b_r w(n-r) \end{aligned} \tag{1.1.15}$$

Bộ phương trình này có thể biểu diễn như trong hình 1.2b, với bộ nhớ để lưu giữ được yêu cầu và chứa các giá trị dãy trễ.

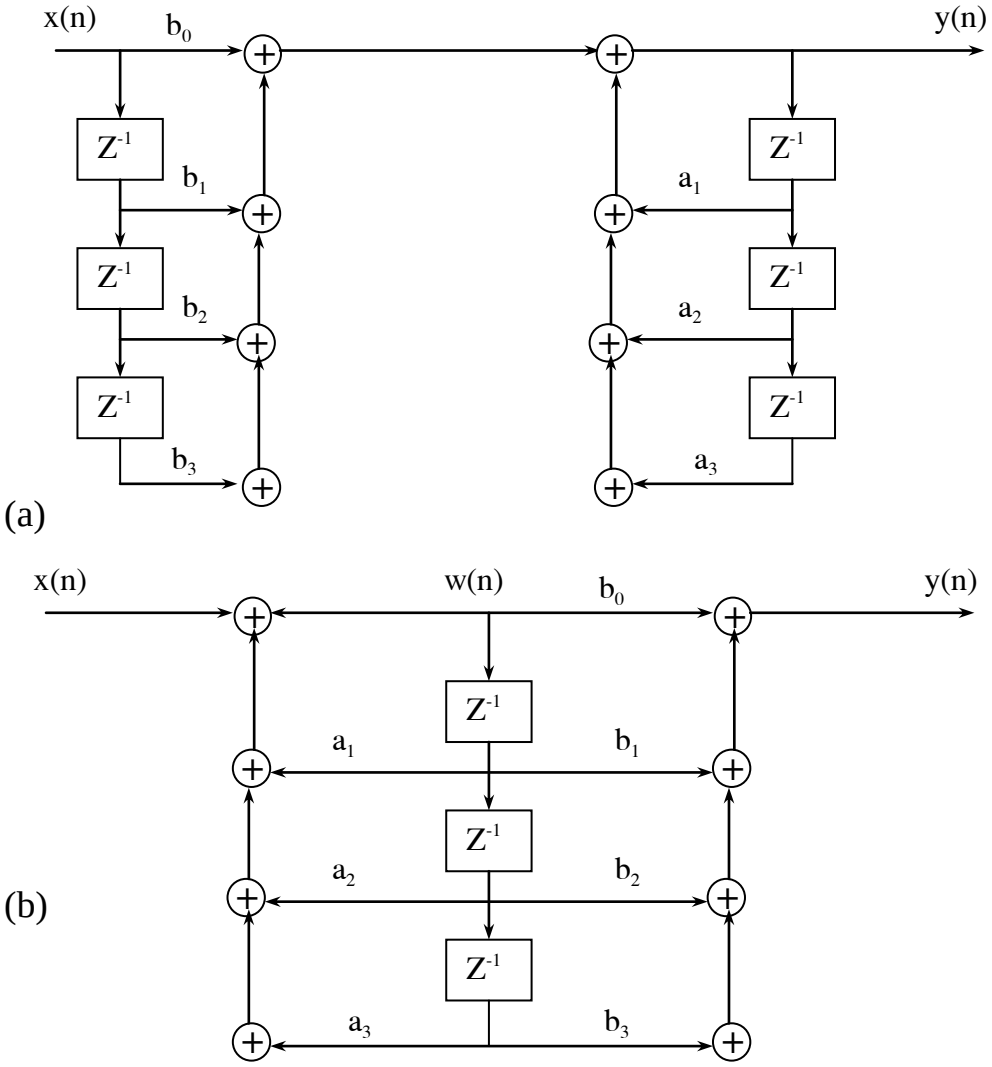
Phương trình (1.1.7) chỉ ra rằng H(Z) có thể biểu diễn như một tích các

điểm cực. Những điểm cực và điểm không này là các cặp liên hiệp phức, vì các hệ số a_k và b_k là thực.

Bằng những nhóm liên hiệp phức điểm cực và điểm không trong cặp liên hợp phức, nó cũng có thể biểu diễn H(Z) như tích của các hàm hệ thống cơ bản cấp hai dạng:

$$H(Z) = A\prod_{k=1}^{K}\left[\frac{1+b_{1k}Z^{-1}+b_{2k}Z^{-2}}{1-a_{1k}Z^{-1}-a_{2k}Z^{-2}}\right] \qquad (1.1.16)$$

K là phần nguyên của (N+1)/2. Hệ thống cấp hai này được biểu diễn như trong hình 1.3a cho trường hợp N=M=4.

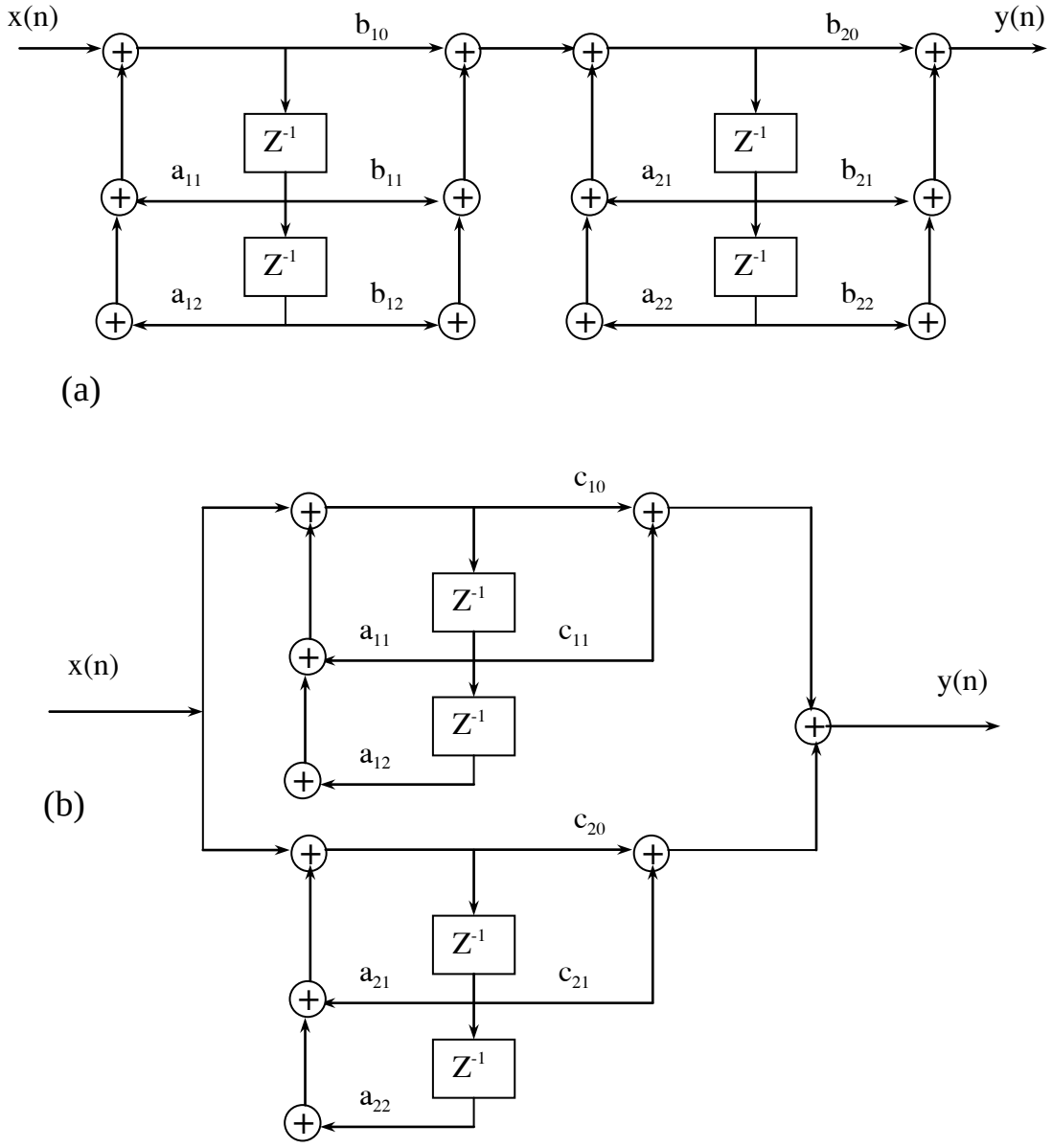


Hình 1.2. (a) Cấu trúc dạng trực tiếp;
(b) Cấu trúc dạng trực tiếp tối giản

Tiếp tục, một cấp độ cao hơn được xét đến. Bằng cách kết hợp những phần liên quan đến cực liên hợp phức, H(Z) có thể viết dạng:

$$H(Z) = \sum_{k=1}^{K} \frac{c_{0k} + c_{1k} Z^{-1}}{1 - a_{1k} Z^{-1} - a_{2k} Z^{-2}} \qquad (1.1.17)$$

Điều này gợi ý một dạng sơ đồ song song biểu diễn như hình 1.3b cho N=4.



Hình 1.3. (a) Dạng tầng;
(b) Dạng song song

Trong những ứng dụng lọc tuyến tính, dạng song song đưa ra những đặc tính cao hơn về phương diện làm tròn giảm tiếng ồn, các sai số hệ số, và tính ổn định.

Chương 2.

BỘ LỌC THÍCH NGHI

2.1. Bộ lọc FIR thích nghi dạng trực tiếp

Từ chuẩn bình phương tối thiểu đưa tới khuôn mẫu chung thiết lập công thức tuyến tính cho hệ số bộ lọc.

$$\sum_{k=0}^{M-1} h(k) r_{cx}(l-k) = r_{dx}(l+D), \qquad l = 0,1,2,\dots\dots,M-1 \tag{2.1.1}$$

Dãy tự tương quan $r_{cx}(l)$ và tương quan chéo $r_{dx}(l)$ nhận được từ dữ liệu, do đó chúng mô tả những ước lượng của dãy tương quan và tự tương quan thực. Hệ số h(k) ở (2.1.1) cũng là những ước lượng của hệ số thực. Độ chính xác của các ước lượng phụ thuộc vào độ dài của bản ghi dữ liệu, đó là 1 vấn đề cần cân nhắc trong hệ thống xử lí của bộ lọc.

Vấn đề thứ 2 cần quan tâm đó là quá trình ngẫu nhiên cơ bản x(n) thường xuyên không ổn định. Ví dụ, trong bộ hiệu chỉnh kênh, các thông số đặc trưng cho tần số có thể biến đổi theo thời gian. Như 1 hệ quả, các dãy tương quan và tự tương quan thống kê, và các ước lượng của chúng thay đổi theo thời gian. Điều này làm cho hệ số của bộ lọc thích nghi cũng phải thay đổi theo thời gian để phản ánh được các thông số thay đổi theo thời gian của tín hiệu ở đầu vào bộ lọc. Điều này cũng kéo theo chất lượng của ước lượng không thể tăng bằng cách đơn giản là tăng số mẫu tín hiệu được sử dụng trong ước lượng các dãy tương quan và tự tương quan.

Có nhiều cách để hệ số của bộ lọc có thể biến đổi theo thời gian cùng với các thông số thống kê theo thời gian của tín hiệu. Phương pháp phổ biến nhất là đưa vào bộ lọc dựa trên các mẫu liên tiếp một cách đệ quy mỗi khi nhận được một mẫu tín hiệu. Cách thứ 2 là ước lượng $r_{cx}(l)$ và $r_{dx}(l)$ trên cơ sở các khối liên tiếp, và không duy trì sự liên tục của các giá trị của hệ số bộ lọc từ một khối dữ liệu tới một khối khác. Kích thước khối phải tương đối nhỏ, chiếm một khoảng thời gian ngắn khi so sánh với khoảng thời gian mà các

đặc trưng thống kê của dữ liệu thay đổi một cách đáng kể.

Khi nghiên cứu về các thuật toán của bộ lọc thích nghi, ta chỉ chú ý tới các thuật toán đệ quy thời gian mà nó cập nhật hệ số dựa trên các mẫu liên tiếp. Trong thực tế ta xét tới hai dạng thuật toán: thuật toán LMS (Least Mean Squares), là thuật toán dựa trên kiểu gradient hướng theo sự thay đổi theo thời gian của các thông số đặc trưng của tín hiệu, và loại thật toán bình phương tối thiểu đệ quy, là thuật toán phức tạp hơn so với LMS.

2.1.1. Tiêu chuẩn lỗi trung bình bình phương tối thiểu (MMES)

Thuật toán LMS được xác định dễ dàng nhất bằng cách lập công thức tối ưu tính hệ số của bộ lọc FIR như một sự ước lượng dựa trên việc tối thiểu hóa lỗi bình phương trung bình.

Ta giả sử có dãy dữ liệu x(n) là các mẫu từ việc xử lí ngẫu nhiên dãy tự tương quan

$$\gamma_{xx}(m) = E[x(n)x^*(n-m)] \tag{2.1.2}$$

Từ những mẫu này ta ước lượng dãy d(n) bằng cách đưa x(n) qua bộ lọc FIR với hệ số bộ lọc h(n), $0 \le n \le M-1$. Đầu ra của bộ lọc là

$$\bar{d}(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \tag{2.1.3}$$

Với $\bar{d}(n)$ là ước lượng của d(n) với lỗi ước lượng là

$$e(n) = d(n) - \bar{d}(n)$$

$$= d(n) - \sum_{k=0}^{M-1} h(k)x(n-k) \tag{2.1.4}$$

Lỗi trung bình phương như là một hàm của hệ số bộ lọc

$$J(h_M) = E[|e(n)^2|]$$

$$= E\left[\left|d(n) - \sum_{k=0}^{M-1} h(k)x(n-k)\right|^2\right]$$
$$= E\left\{|d(n)|^2 - 2Re\left[\sum_{l=0}^{M-1} h^*(l)d(n)x^*(n-l)\right]\right.$$
$$\left. + \sum_{k=0}^{M-1}\sum_{l=0}^{M-1} h^*(l)h(k)x^*(n-l)x(n-k)\right\}$$
$$= {\sigma^2}_d - 2Re\left[\sum_{l=0}^{M-1} h^*(l)\gamma_{dx}(l)\right] + \sum_{k=0}^{M-1}\sum_{k=0}^{M} h^*(l)h(k)\gamma_{xx}(l-k)$$

(2.1.5)

Với ${\sigma^2}_d = E[|d(n)^2|]$ và h_M là vector hệ số.

${h^*}_M$ là liên hợp của h_M

${h'}_M$ là chuyển vị của h_M

Ta thấy rằng MSE là hàm bậc 2 của hệ số bộ lọc. Do đó giá trị nhỏ nhất của $J(h_M)$ dẫn tới việc thiết lập biểu thức tuyến tính M

$$\sum_{k=0}^{M-1} h(k)\,\gamma_{xx}(l-k) = \gamma_{dx}(l) \qquad l = 0,1,\dots\dots\dots\dots,M-1$$

(2.1.6)

Bộ lọc có hệ số nhận được từ (2.1.6) (2.1.6 là công thức Wiener-Hopf) được gọi là bộ lọc Wiener.

Khi so sánh (2.1.6) và (2.1.1) ta thấy rằng chúng cùng dạng. Ở (2.1.1) ta dùng sự ước lượng về tự tương quan và tương quan chéo để xác định hệ số bộ lọc, trong khi ở (2.1.6) người ta dùng dãy tự tương quan và tương quan chéo thống kê được, vì thế (2.1.6) cung cấp hệ số bộ lọc tối ưu trong hướng MSE, trong khi (2.1.1) đưa ra sự ước lượng về hệ số tối ưu.

Biểu thức (2.1.6) ở dạng ma trận như sau :

$$\Gamma_M h_M = \gamma_d \qquad (2.1.7)$$

Với Γ_M là ma trận Toeplizt $(= M \times M)$ với thành phần $\Gamma_{lk} = \gamma_{cx}(l-k)$ và γ_d bằng $M \times 1$ vetor tương quan chéo với thành phần $\gamma_{dx}(l), l - 0,1,\dots\dots M-1$. Và ta có hệ số bộ lọc tối ưu là

$$h_{opt} = {\Gamma_M}^{-1}\gamma_d \tag{2.1.8}$$

Và

$$J_{min} = J(h_{opt}) = {\sigma^2}_d - \sum_{k=0}^{M-1} h_{opt}(k){\gamma^*}_{dx}(k) = {\sigma^2}_d - \gamma_d^H {\Gamma_M}^{-1}\gamma_d \tag{2.1.9}$$

Với H là chuyển vị liên hợp.

Việc thiết lập biểu thức tuyến tính (2.1.6) cũng có thể thực hiện bằng cách đưa ra nguyên lí trực giao trong việc ước lượng trung bình bình phương. Theo nguyên lí này, lỗi ước lượng trung bình bình phương được tối thiểu hóa khi e(n) trực giao với ước lượng $\bar{d}(n)$

$$E\left[e(n)\bar{d}^*(n)\right] = 0 \tag{2.1.10}$$

$$E\left[\sum_{k=0}^{M-1} h(k)e(n)x^*(n-k)\right] = \sum_{k=0}^{M-1} h(k)E[e(n)x^*(n-k)] = 0$$

Hoặc tương đương với

$$E[e(n)x^*(n-l)] = 0 \qquad l = 0,1,\dots\dots\dots\dots\dots,M-1 \tag{2.1.11}$$

Nếu ta thay thế e(n) trong (2.1.11) bằng e(n) trong (2.1.4) và sử dụng phép toán trung bình ta nhận được biểu thức như (2.1.6).

Do $\bar{d}(n)$ là trực giao với e(n), lỗi bình phương trung bình nhỏ nhất là

$$J_{min} = E\left[e(n)\bar{d}^*(n)\right] = E[|d(n)|^2] - \sum_{k=0}^{M-1} h_{opt}(k)\gamma_{dx}^*(k) \tag{2.1.12}$$

Hệ số bộ lọc tối ưu như ở (2.1.8) có thể được thực hiện một cách hiệu quả khi dùng thuật toán Levinson-Durbin. Tuy nhiên ta cần chú ý tới việc dùng phương pháp gradient, việc đó dẫn tới thuật toán LMS cho bộ lọc.

2.1.2. Thuật toán Widrow LMS

Có nhiều phương pháp để thiết lập biểu thức tuyến tính (2.1.6) hay (2.1.7) cho hệ số bộ lọc tối ưu. Ở đây ta xét tới phương pháp đệ quy, nó cho phép tìm cực tiểu của một hàm nhiều biến, MSE là một hàm bậc 2 của hệ số bộ lọc, do vậy hàm này có duy nhất một giá trị cực tiểu và chúng ta sẽ xác

định nó bằng cách lặp nhiều lần.

Ta giả thiết ma trận tự tương quan Γ_M và vector tương quan chéo γ_d đã biết trước, do đó $J(h_M)$ là hàm đã biết của hệ số h(n), $0 \leq n \leq M-1$. Các thuật toán để tính toán một cách đệ quy hệ số bộ lọc và tìm cực tiểu của $J(h_M)$ có dạng:

$$h_M(n+1) = h_M(n) + !\Delta(n)S(n), \qquad n = 0,1,\dots\dots\dots \tag{2.1.13}$$

Với $h_M(n)$ là vector của hệ số bộ lọc tại bước lặp thứ n

$\Delta(n)$ là độ lớn bước nhảy tại bước lặp thứ n

$S(n)$ là vector hướng cho bước lặp thứ n

giá trị ban đầu $h_M(0)$ được chọn tùy ý.

Phương pháp đơn giản nhất để tìm cực tiểu của $J(h_M)$ một cách đệ quy là dựa vào việc tìm theo sự hạ thấp của đường dốc, ở phương pháp này vector $S(n) = -g(n)$, với $g(n)$ là vector gradient tại bước nhảy thứ n.

$$g(n) = \frac{dJ(h_M(n))}{dh_M(n)} = 2[\Gamma_M h_M(n) - \gamma_d, \qquad n = 0,1,\dots\dots \tag{2.1.14}$$

Do đó ta sẽ tính vector gradient cho mỗi bước nhảy và thay đổi giá trị của $h_M(n)$ theo gradient chiều ngược, và ta có thuật toán đệ quy dựa trên phương pháp tìm theo sự hạ thấp của đường dốc là:

$$h_M(n+1) = h_M(n) - !\Delta(n)g(n) \tag{2.1.15}$$

Tương đương với

$$h_M(n+1) = [I - \Delta(n)\Gamma_M]h_M(n) + \Delta(n)\gamma_d \tag{2.1.16}$$

Ta không chứng minh thuật toán dẫn tới việc $h_M(n)$ hộ tụ tới h_{opt} khi $n \to \infty$, dãy độ lớn bước nhảy $\Delta(n)$ hoàn toàn khả tổng và $\Delta(n) \to 0$ khi $n \to \infty$.

Một số thuật toán khác cho ta sự hội tụ nhanh hơn như thuật toán liên hợp gradient và thuật toán Fletcher-Powel. Trong thuật toán liên hợp gradient:

$$S(n) = \beta(n-1)S(n-1) - g(n) \tag{2.1.17}$$

Với $\beta(n)$ là hàm vô hướng của vector gradient

Trong thuật toán Fletcher-Powel:

$$S(n) = -H(n)g(n) \tag{2.1.18}$$

Với $H(n)$ là ma trận dương $M \times M$ và nó hội tụ ngược với Γ_M.

Rõ ràng 3 thuật toán có cách xác định hướng vector khác nhau.

Ba thuật toán trên là thích hợp khi Γ_M và γ_d đã biết, tuy nhiên đó không phải là trường hợp trong các ứng dụng của bộ lọc thích nghi. Khi không biết Γ_M và γ_d ta có thể thay thế $S(n)$ ước lượng cho $S(n)$ thực tế.

Đầu tiên, chú ý rằng vecter gradient ở (2.1.14) cũng có thể được thể hiện ở điều kiện trực giao như trong (2.1.10), thực tế (2.1.10) tương đương với:

$$E[e(n)X_M^*(n) = \gamma_d - \Gamma_M h_M(n) \tag{2.1.19}$$

Với $X_M(n)$ là vector với các thành phần $x(n-l), l = 0,1, \dots\dots\dots, M-1$. Do vậy vector gradient là

$$g(n) = -2E[e(n)X_M^*(n) \tag{2.1.20}$$

Từ (2.1.20) ta có ước lượng khá chính xác về vector gradient

$$g'(n) = -2e(n)X_M^*(n) \tag{2.1.21}$$

Với $e(n) = d(n) - \bar{d}(n)$ và $X_M(n)$ là bộ mẫu tín hiệu M trong bộ lọc ở bước lặp thứ n, khi thay $\bar{g}(n)$ cho $g(n)$ ta có thuật toán

$$h_M(n+1) = h_M(n) + \Delta(n)e(n)X_M^*(n) \tag{2.1.22}$$

Và nó gọi là thuật toán hạ bậc gradient ngẫu nhiên, thuật toán này được áp dụng phổ biến trong các bộ lọc thích nghi để sử dụng thuật toán độ lớn bước cố định vì hai lí do. Một là thuật toán độ lớn bước cố định được thực hiện dễ dàng với cả phần cứng và phần mềm. Thứ hai, một bước nhảy đã ấn định kích thước thì thích ứng với dòng tín hiệu thay đổi theo thời gian, trong khi nếu $\Delta(n) \to 0$ khi $n \to \infty$, việc thích nghi với sự thay đổi của tín hiệu không thể xảy ra. Vì những lí do đó (2.1.22) có thể được viết

$$h_M(n+1) = h_M(n) + \Delta e(n)X_M^*(n) \tag{2.1.23}$$

Với Δ là kích thước bước nhảy đã được ấn định.

Thuật toán này được đưa ra đầu tiên bởi Windrow và Hoft (1960), giờ đây nó được biết đến rộng rãi với cái tên thuật toán LMS (Least Mean Square). Rõ ràng, nó là thuật toán gradient ngẫu nhiên.

Thuật toán LMS là thuật toán sử dụng dễ dàng, vì thế nó được dùng rộng rãi trong nhiều ứng dụng của bộ lọc thích nghi. Các thuộc tính và giới hạn của nó được nghiên cứu kĩ lưỡng. Trong phần dưới đây, ta sẽ đưa ra bản tóm tắt về các thuộc tính quan trọng của nó liên quan tới sự hội tụ, độ ổn định và nhiễu do việc ước lượng vector gradient. Sau đó ta sẽ so sánh thuộc tính của nó với các thuật toán bình phương tối thiểu đệ quy phức tạp hơn.

Nhiều biến dạng của thuật toán LMS cơ bản được đặt ra trên lí thuyết và được thực hiện trong một vài ứng dụng của bộ lọc, một trong số đó là: nếu ta lấy trung bình các vector gradient qua nhiều lần lặp để điều chỉnh hệ số bộ lọc, ví dụ trung bình K vector gradient là

$$\bar{g}'(nK) = \frac{-2}{K}\sum_{k=0}^{K-1} e(nK+k)X_M^*(nK+k) \tag{2.1.24}$$

Và theo công thức đệ quy, việc thiết lập hệ số bộ lọc ở mỗi bước lặp K là

$$h_M\big((n+1)K\big) = h_M(nK) - !\Delta\bar{g}'(nK) \tag{2.1.25}$$

Việc lấy trung bình như ở (2.1.24) giảm nhiễu trong việc ước lượng vector gradient.

Một cách khác là đặt một bộ lọc thông thấp và dùng đầu ra của nó để ước lượng vector gradient. Ví dụ, một bộ lọc thông thấp đơn giản cung cấp vector gradient ở đầu ra

$$S'(n) = \beta S'(n-1) - g'(n), \quad S(0) = -g(0) \tag{2.1.26}$$

Với $0 \le \beta \le 1$ xác định dải thông của bộ lọc thông thấp. Khi β tiến tới 1, dải thông bộ lọc nhỏ và việc lấy trung bình được thực hiện trên rất nhiều vector gradient. Mặt khác, khi β nhỏ bộ lọc có dải thông lớn và do đó ít vector gradient được lấy trung bình hơn. Với $g'(n)$ ở (2.1.26) ta nhận được một phiên bản mới của thuật toán LMS

$$h_M(n+1) = h_M(n) + !\Delta S'(n)$$

(2.1.27)

2.1.3. Thuộc tính của thuật toán LMS

Trên thực tế ta tập trung vào thuộc tính hộ tụ, tính ổn định và việc xử lí nhiễu phát sinh khi thay thế vector gradient nhiễu cho vector gradient thực. Việc ước lượng nhiễu của vector gradient làm cho hệ số bộ lọc dao động ngẫu nhiên, và do đó việc giải thích thuộc tính của thuật toán được thực hiện bằng cách thống kê.

Tính hội tụ và ổn định của thuật toán LMS được nghiên cứu bằng việc xác định cách mà giá trị trung bình của $h_M(n)$ hội tụ tới hệ số tối ưu h_{opt}

$$\begin{aligned}\bar{h}_M(n+1) &= \bar{h}_M(n) + \Delta E[e(n)X_M^*(n)] \\ &= \bar{h}_M(n) + \Delta\left[\gamma_d - \Gamma_M \bar{h}_M(n)\right] \\ &= (I - \Delta\Gamma_M)\bar{h}_M(n) + \Delta\gamma_d\end{aligned}$$

(2.1.28)

Với $\bar{h}_M(n) = E[h_M(n)]$ và I là ma trận đồng nhất.

Hệ thức đệ quy (2.1.28) được thể hiện bởi hệ thống điều khiển vòng kín như ở hình 2.1. Tốc độ hội tụ và tính ổn định của hệ thống này được điều khiển bằng cách chọn kích cỡ bước nhảy Δ. Để xác định trạng thái hội tụ thuận tiện nhất là tách rời M phương trình sai phân đồng thời cho ở (2.1.28) bằng cách sử dụng phương pháp biến đổi tuyến tính vector hệ số trung bình $\bar{h}_M(n)$. Khi chú ý tới ma trận tự tương quan Γ_M, ta có biến đổi tương ứng

$$\Gamma_M = UAU^H$$

(2.1.29)

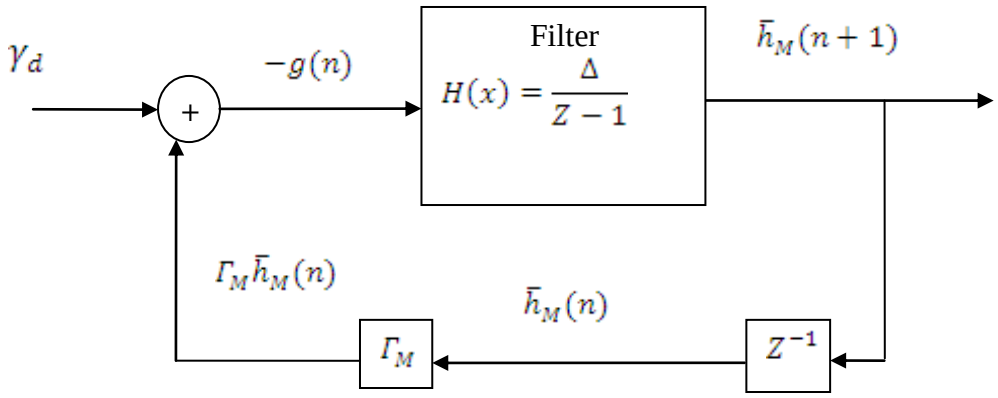
Với U là ma trận chuẩn hóa của Γ_M và A là đường chéo của ma trận với các thành phần $\lambda_k, 0 \leq k \leq M-1$, bằng với giá trị riêng của Γ_M

Thay (2.1.29) vào (2.1.28) ta có

$$\bar{h}_M^0(n+1) = (I - \Delta A)\bar{h}_M^0(n) + \Delta\gamma_d^0$$

(2.1.30)

Với $\bar{h}_M^0(n) = U^H h_M(n)$ và $\gamma_d^0 = U^H\gamma_d$



Hình 2.1 Hệ thống điều khiển kín

Tính hội tụ và ổn định được xác định từ công thức đồng nhất

$$\bar{h}_M^0(n+1) = (I - \Delta A)\bar{h}_M^0(n) \tag{2.1.31}$$

Ta có

$$h^0(k,n) = C(1 - \Delta\lambda_{k)}{}^n u(n), \qquad k = 0,2,\dots\dots M-1 \tag{2.1.32}$$

Với C là hằng số tùy ý

$u(n)$ là dãy bước nhảy đơn vị

Rõ ràng $h^0(k,n)$ hội tụ tới 0 khi

$|1 - \Delta\lambda_k| < 1$

Tương đương với

$$0 < \Delta < \frac{2}{\lambda_k}, \quad k = 0,1,\dots\dots M-1 \tag{2.1.33}$$

Tốc độ hội tụ cực đại khi $\Delta = {}^1/_{\lambda_k}$.

Điều kiện ở (2.1.33) cho sự hội tụ của phương trình sai phân đồng nhất đối với hệ số bộ lọc thứ k (mô hình thứ k của hệ thống kín) phải thỏa mãn cho mọi k=0, 1, ..., M-1. Do vậy dải giá trị của Δ đảm bảo sự hội tụ của vector hệ số trong thuật toán LMS là

$$0 < \Delta < \frac{2}{\lambda_{max}} \tag{2.1.34}$$

Với λ_{max} là giá trị riêng lớn nhất của Γ_M

Do Γ_M là một ma trận tự tương quan, giá trị riêng của nó không âm. Do vậy cận trên của λ_{max} là

$$\lambda_{max} < \sum_{k=0}^{M-1} \lambda_k \, trace\, \Gamma_M = M\gamma_{xx}(0)$$

(2.1.35)

Với $\gamma_{xx}(0)$ là nguồn tín hiệu đầu vào, nó dễ dàng được ước lượng từ tín hiệu nhận được, do vậy cận trên của Δ là $2/M\gamma_{xx}(0)$.

LMS hội tụ nhanh khi $|1 - \Delta\lambda_k|$ nhỏ. Tuy nhiên, ta không thể có điều kiện như mong muốn và vẫn thỏa mãn cận trên khi có một khoảng cách lớn giữa giá trị riêng lớn nhất và nhỏ nhất của Γ_M. Nói cách khác, nếu ta chọn Δ bằng $1/\lambda_{max}$, tốc độ hội tụ của LMS sẽ được xác định bởi sự suy giảm của mô hình tương ứng tới giá trị nhỏ nhất λ_{min}. Ở mô hình này, thay $\Delta = 1/\lambda_{max}$ vào công thức (2.1.32) ta có

$$\bar{h}_M^0(k,n) = C\left(1 - \frac{\lambda_{min}}{\lambda_{max}}\right)^n u(n)$$

Tỉ số $\lambda_{min}/\lambda_{max}$ giới hạn tốc độ hội tụ, nếu $\lambda_{min}/\lambda_{max}$ nhỏ ($\ll 1$) sự hội tụ sẽ chậm và ngược lại khi $\lambda_{min}/\lambda_{max} \to 1$.

Một đặc tính quan trọng nữa của LMS là nhiễu do việc sử dụng ước lượng của vector gradient. Nhiễu này làm cho hệ số bộ lọc dao động ngẫu nhiên quanh giá trị tối ưu và điều đó làm tăng giá trị cực tiểu của MSE ở đầu ra của bộ lọc. Do đó tổng MSE là $J_{min} + J_\Delta$ với J_Δ là lỗi bình phương trung bình dư.

Tổng MSE ở đầu ra bộ lọc có thể được viết như sau:

$$J\big(h_M(n)\big) = J_{min} + \big(h_M(n) - h_{opt}\big)\Gamma_M\big(h_M(n) - h_{opt}\big)^*$$

(2.1.36)

Với h_{opt} là hệ số tối ưu của bộ lọc được xác định bởi (2.1.8)

$J\big(h_M(n)\big)$ được gọi là *đường cong tiếp thu*

Khi thay Γ_M như ở (6.2.29) và biến đổi trực giao tuyến tính ta có

$$J\big(h_M(n)\big) = J_{min} + \sum_{k=0}^{M-1} \lambda_k \left|h^0(k,n) - h_{opt}^0(k)\right|^2$$

(2.1.37)

Với $h^0(k,n) - h^0_{opt}(k)$ được coi là lỗi trong hệ số bộ lọc thứ k (trong hệ thống sắp xếp trực giao). Và lỗi bình phương trung bình dư là

$$J_\Delta = \sum_{k=0}^{M-1} \lambda_k E[\left|h^0(k,n) - h^0_{opt}(k)\right|^2]$$

(2.1.38)

Ta giả sử giá trị trung bình của hệ số bộ lọc $h_M(n)$ hộ tụ tới giá trị tối ưu của nó là h_{opt}. Và phần $\Delta e(n)X^*_M(n)$ trong (2.1.23) là vector nhiễu trung bình không. Hiệp phương sai của nó là

$$cov[\Delta e(n)X^*_M(n)] = \Delta^2 E[|e(n)|^2 X_M(n)X^H_M(n)$$

(2.1.39)

Ta giả sử $|e(n)|^2$ không liên quan tới vector tín hiệu, dù giả thiết này không chặt chẽ lắm nhưng nó rút ngắn dẫn dắt và cho kết quả đầy đủ. Và

$$cov[\Delta e(n)X^*_M(n)] = \Delta^2 E[|e(n)|^2]E[X_M(n)X^H_M(n)]$$
$$= \Delta^2 J_{min}\Gamma_M$$

(2.1.40)

Đối với vector hệ số $h^0_M(n)$, cộng thêm nhiễu, ta có

$$h^0_M(n+1) = (I - \Delta A)h^0_M(n) + \Delta\gamma^0_d + w^0_M(n)$$

(2.1.41)

Với $w^0_M(n)$ là vector nhiễu cộng thêm, nó liên quan tới vector nhiễu $\Delta e(n)X^*_M(n)$ qua biến đổi

$$w^0(n) = U^H[\Delta e(n)X^*_M(n)]$$
$$= \Delta e(n)U^H X^*_M(n)$$

(2.1.42)

Có thể thấy ma trận hiệp phương sai của vector nhiễu là

$$cov[w^0(n)] = \Delta^2 J_{min}U^H\Gamma_M U = \Delta^2 J_{min}A$$

(2.1.43)

Do vậy các thành phần M của $w^0_M(n)$ không liên quan tới nhau và mỗi thành phần có một sai số $\sigma^2_k = \Delta^2 J_{min}\lambda_k, k = 0,1,\dots\dots\dots M-1.$

Do các thành phần M của $w^0_M(n)$ không liên quan tới nhau nên ta có thể tách riêng M công thức, mỗi công thức bậc nhất thể hiện một bộ lọc với đáp ứng xung $(1-\Delta\lambda_k)^n$. Khi một bộ lọc bị ảnh hưởng bởi dãy nhiễu $w^0_k(n)$,

nhiễu ở đầu ra của bộ lọc là

$$E\left[\left|h^0(k,n)-h^0_{opt}(k)\right|^2\right]=$$
$$\sum_{n=0}^{\infty}\sum_{m=0}^{\infty}(1-\Delta\lambda_k)^n(1-\Delta\lambda_k)^m E[w_k^0(n)w_k^{0*}(n)]$$

(2.1.44)

Ta giả thiết $w_k^0(n)$ là nhiễu trắng, và (2.1.44) được rút gọn

$$E\left[\left|h^0(k,n)-h^0_{opt}(k)\right|^2\right]=\frac{\sigma_k^2}{1-(1-\Delta\lambda_k)^2}=\frac{\Delta^2 J_{min}\lambda_k}{1-(1-\Delta\lambda_k)^2}$$

(2.1.45)

Thay (2.1.45) vào (2.1.38) ta có

$$J_\Delta=\Delta^2 J_{min}\sum_{k=0}^{M-1}\frac{\lambda_k^2}{1-(1-\Delta\lambda_k)^2}$$

(2.1.46)

Khi coi $\Delta\lambda_k \ll 1$ với mọi k, ta được

$$J_\Delta \approx \Delta^2 J_{min}\sum_{k=0}^{M-1}\frac{\lambda_k^2}{2\Delta\lambda_k}$$
$$\approx \frac{1}{2}\Delta J_{min}\sum_{k=0}^{M-1}\lambda_k$$
$$\approx \frac{\Delta M J_{min}\gamma_{xx}(0)}{2}$$

(2.1.47)

Với $\gamma_{xx}(0)$ là công suất tín hiệu vào.

Ta thấy lỗi bình phương trung bình dư J_Δ thì tỉ lệ thuận với bước nhảy Δ. Do đó khi chọn Δ phải đảm bảo hội tụ nhanh và lỗi bình phương trung bình dư nhỏ. Trên thực tế, mong muốn có $J_\Delta < J_{min}$, ta có

$$\frac{J_\Delta}{J_{min}}\approx\frac{\Delta M\gamma_{xx}(0)}{2}<1$$

Tương đương

$$\Delta<\frac{2}{M\gamma_{xx}(0)}$$

(2.1.48)

Trong điều kiện ổn định Δ phải thỏa mãn (2.1.48). Nói cách khác, lỗi bình phương trung bình dư cũng làm giảm đáng kể chất lượng bộ lọc thích nghi.

Những lí giải về lỗi bình phương trung bình dư ở trên là dựa vào giả thiết giá trị trung bình của hệ số bộ lọc hội tụ tới giá trị tối ưu h_{opt}. Ở điều kiện đó, kích thước bước nhảy Δ phải thỏa mãn (2.1.48). Mặt khác, ta đã xác định để vector hệ số trung bình hội tụ thì điều kiện cần là $\Delta < 2/\lambda_{max}$. Trong khi việc chọn Δ gần với cận trên có thể dẫn tới sự hội tụ ban đầu của thuật toán gradient, khi mở rộng Δ sẽ làm thuật toán gradient LMS ngẫu nhiên mất ổn định.

Tính hội tụ ban đầu hay trạng thái nhất thời của LMS được nhiều nhà khoa học nghiên cứu. Họ chỉ ra rằng kích thước bước nhảy tỉ lệ thuận với độ dài bộ lọc thích nghi. Cận trên (2.1.48) là cần thiết để đảm bảo sự hội tụ ban đầu của LMS gradient ngẫu nhiên. Thực tế thường chọn $\Delta < 1/M\gamma_{xx}(0)$.

Trong hoạt động của LMS, việc chọn kích thước bước nhảy quan trọng hơn. Ta có thể giảm lỗi bình phương trung bình dư bằng cách giảm Δ tới điểm mà tại đó tổng của lỗi bình phương trung bình đầu ra giảm. Điều đó xảy ra khi các thành phần gradient $e(n)x^*(n-l), l = 0,1,\dots\dots M-1$ được ước lượng, sau phép nhân bởi thông số độ lớn bậc nhỏ Δ (nhỏ hơn một nửa của bit nhỏ nhất trong biểu diễn điểm cố định của hệ số bộ lọc). Do đó điều quan trọng là kích thước bước nhảy phải đủ rộng để hệ số bộ lọc hội tụ tới h_{opt}. Nếu muốn giảm kích thước bước nhảy một cách đáng kể thì điều cần thiết là phải tăng độ chính xác của hệ số bộ lọc. Thông thường, 16 bits được dùng cho các hệ số bộ lọc, với từ 8 đến 12 bits dùng cho xử lí số học trong lọc dữ liệu, từ 4 đến 8 bits cho xử lí thích nghi. Các thành phần gradient ước lượng dùng số bit ít nhất.

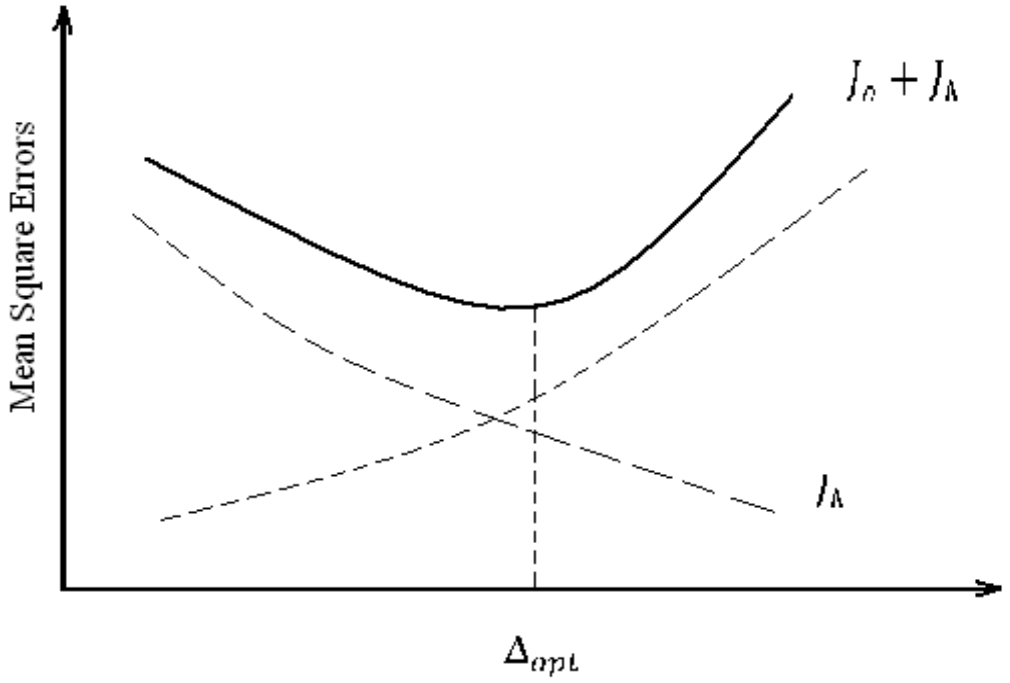
Cuối cùng, ta cần chỉ ra rằng thuật toán LMS thích ứng với dòng tín hiệu thống kê biến đổi chậm theo thời gian, như trong trường hợp cực tiểu MSE và hệ số tối ưu biến đổi theo thời gian. Nói cách khác, $J_{min}(n)$ là một hàm theo thời gian. LMS chứa một loại lỗi khác, đó là lỗi trễ, là lỗi giá trị bình phương trung bình giảm cùng với việc tăng kích thước bước nhảy. Tổng lỗi MSE giờ là

$$J_{total} = J_{min}(n) + J_{\Delta} + J_l$$

(2.1.49)

Nếu ta vẽ J_Δ và J_l như một hàm của Δ, ta có hình 2.2. Ta thấy khi Δ tăng thì J_Δ tăng còn J_l lại giảm, từ đó thấy giá trị Δ mà tại đó tổng lỗi là nhỏ nhất.

Khi tín hiệu biến đổi nhanh theo thời gian lỗi trễ sẽ lấn át chất lượng bộ lọc. như trong trường hợp $J_l > J_{min} + J_\Delta$, khi giá trị lớn nhất của Δ được dùng. Khi đó thuật toán LMS không còn thích hợp cho các ứng dụng và cần tới một thuật toán phức tạp hơn, thuật toán bình phương tối thiểu đệ quy, để có được sự hội tụ nhanh hơn và bám sát.



Hình 2.2 Lỗi trung bình bình phương dư J_Δ và lỗi trễ J_l

2.1.4. Thuật toán bình phương tối thiểu đệ quy

Lợi thế cơ bản của LMS là cách tính toán đơn giản. Tuy nhiên, nó lại hội tụ chậm đặc biệt khi các giá trị riêng của ma trận tự tương quan Γ_M có khoảng cách lớn. Nhìn theo quan điểm khác, thuật toán LMS chỉ có một thông số để điều khiển tốc độ hội tụ, đó là Δ. Do Δ bị hạn chế bởi cận trên để đảm bảo tính ổn định, các giá trị riêng nhỏ hơn nên hội tụ rất chậm.

Để có được sự hội tụ nhanh hơn, cần có một thuật toán hoàn chỉnh hơn cho nhiều thông số hơn. Thực tế, nếu ma trận tự tương quan có các giá trị riêng không bằng nhau $\lambda_0, \lambda_1, \ldots\ldots\ldots, \lambda_{M-1}$, ta phải dùng một thuật toán có

M thông số, mỗi thông số cho một giá trị riêng.

Để dẫn tới các thuật toán cho sự hội tụ nhanh hơn, ta cần chấp nhận thay thế phép xấp xỉ thống kê dựa trên chuẩn MSE bằng chuẩn bình phương tối thiểu. Ta sẽ quan tâm trực tiếp tới dữ liệu $x(n)$ và nhận được ước lượng về tương quan từ dữ liệu.

Điều thuận lợi để thể hiện thuật toán bình phương tối thiểu là dạng ma trận, các thuật toán đệ quy trong miền thời gian. Cũng cần phải đưa chỉ số thời gian và vector hệ số bộ lọc dãy lỗi. Vector hệ số bộ lọc ở miền thời gian n là

$$h_M(n) = \begin{bmatrix} h(0,n) \\ h(1,n) \\ h(2,n) \\ \vdots \\ h(M-1,n) \end{bmatrix} \tag{2.1.50}$$

Với chỉ số M là độ dài bộ lọc. Tương tự, vector tín hiệu đầu vào của bộ lọc là

$$X_M(n) = \begin{bmatrix} x(n) \\ x(n-1) \\ x(n-2) \\ \vdots \\ c(n-M+1) \end{bmatrix} \tag{2.1.51}$$

Giả sử $x(n) = 0$ với n<0. Điều này được gọi là lấy dữ liệu vào qua cửa sổ.

Bình phương tối thiểu đệ quy giờ tính toán như sau: Giả sử ta đã có vector $X_M(l), l = 0,1, \dots \dots \dots . . n$ và ta muốn xác định vector hệ số $h_M(n)$ sao cho nó làm giảm tối thiểu độ lớn của lỗi bình phương.

$$\xi_M = \sum_{l=0}^{n} w^{n-l} \, |e_M(l,n)|^2 \tag{2.1.52}$$

Với lỗi được định nghĩa là khoảng cách giữa dãy mong muốn $d(l)$ và dãy ước lượng $d'(l,n)$

$$\begin{aligned} e_M(l,n) &= d(l) - d'(l,n) \\ &= d(l) - h'_M(n) X_M(l) \end{aligned}$$

(2.1.53)

Với w là chỉ số và $0 < w < 1$.

Chỉ số w là để xử lí hầu hết các điểm dữ liệu mới và do đó cho phép hệ số bộ lọc đáp ứng được các thông số đặc trưng biến đổi theo thời gian của dữ liệu. điều đó được thực hiện bằng cách sử dụng hệ số trọng số lũy thừa với dữ liệu chuyển qua. Tương tự, ta có thể sử dụng cửa sổ trượt độ dài hữu hạn với trọng số đồng dạng trên toàn kích thước cửa sổ. Ta có

$$\overline{N} = \frac{\sum_{n=0}^{\infty} n w^n}{\sum_{n=0}^{\infty} w^n} = \frac{w}{1-w}$$

(2.1.54)

Với $\overline{N}$ là kích thước cửa sổ trượt.

Việc tối thiểu hóa $\boldsymbol{\xi}_M$ mà vẫn ổn định vector hệ số bộ lọc $h_M(n)$ dẫn tới thiết lập công thức tuyến tính

$$R_M(n)\, h_M(n) = D_M(n)$$

(2.1.55)

Với $R_M(n)$ là ma trận tương quan tín hiệu

$$R_M(n) = \sum_{l=0}^{n} w^{n-1} X_M^*(l) X_M'(l)$$

(2.1.56)

$D_M(n)$ là vector tương quan chéo

$$D_M(n) = \sum_{l=0}^{n} w^{n-1} X_M^*(l) d(l)$$

(2.1.57)

Từ (2.1.55) có

$$h_M(n) = R_M^{-1}(n) D_M(n)$$

(2.1.58)

Rõ ràng ma trận $R_M(n)$ giống ma trận tự tương quan Γ_M trong khi ma trận $D_M(n)$ giống vector tương quan chéo γ_d. Tuy nhiên cần nhấn mạnh rằng $R_M(n)$ không phải là ma trận Toeplizt như Γ_M. Ta cũng cần chú ý tới giá trị nhỏ của n, không tính được đảo của $R_M(n)$. Như trong trường hợp cộng thêm ma trận δI_M

vào $R_M(n)$, với I_M là ma trận đồng nhất và δ là hằng số dương nhỏ.

Giả sử ta có (2.1.58) ở (n-l) (ví dụ ta có $h_M(n-l)$) và ta muốn tính $h_M(n)$, do đó trong thực tế không thể thiết lập các biểu thức tuyến tính M cho mỗi thành phần tín hiệu mới. Thay vào đó ta có thể tính ma trận và vector một cách đệ quy. Đầu tiên, tính $R_M(n)$

$$R_M(n) = wR_M(n-1) + X_M^*(n)X_M'(n) \tag{2.1.59}$$

Ta gọi (2.1.59) là biểu thức cập nhật thời gian cho $R_M(n)$.

Do đảo của $R_M(n)$ là cần thiết, ta dùng bổ đề đảo ma trận

$$R_M^{-1}(n) = \frac{1}{w}\left[R_M^{-1}(n-1) - \frac{R_M^{-1}(n-1)X_M^*(n)X_M'(n)R_M^{-1}(n-1)}{w + X_M^*(n)X_M'(n)R_M^{-1}(n-1)}\right] \tag{2.1.60}$$

Ta đặt $P_M(n) = R_M^{-1}(n)$ để thuận tiện cho việc xác định vector khuếch đại Kalman

$$K_M(n) = \frac{1}{w + \mu_M(n)}P_M(n-1)X_M^*(n) \tag{2.1.61}$$

Với $\mu_M(n)$ vô hướng

$$\mu_M(n) = X_M'(n)P_M(n-1)X_M^*(n) \tag{2.1.62}$$

Khi đó (2.1.60) trở thành

$$P_M(n) = \frac{1}{w}[P_M(n-1) - K_M(n)X_M'(n)P_M(n-1)] \tag{2.1.63}$$

Nhân (2.1.63) với $X_M^*(n)$ ta có

$$\begin{aligned} P_M(n)X_M^*(n) &= \frac{1}{w}[P_M(n-1)X_M^*(n) - K_M(n)X_M'(n)P_M(n-1)X_M^*(n)] \\ &= \frac{1}{w}\{[w + \mu_M(n)]K_M(n) - K_M(n)\mu_M(n)\} = K_M(n) \end{aligned} \tag{2.1.64}$$

Do vậy vector khuếch đại Kalman cũng được định nghĩa như $P_M(n)X_M^*(n)$.

Ta dùng ma trận đảo để lập biểu thức tính hệ số bộ lọc một cách đệ quy.

Do

$$h_M(n) = P_M(n)D_M(n) \tag{2.1.65}$$

Và

$$D_M(n) = wD_M(n-1) + d(n)X_M^*(n) \tag{2.1.66}$$

Ta có

$$\begin{aligned} h_M(n) &= \frac{1}{w}[P_M(n-1) - K_M(n)X_M'(n)P_M(n-1)] \\ &\times [wD_M(n-1) + d(n)X_M^*(n)] \\ &= P_M(n-1)D_M(n-1) + \frac{1}{w}d(n)P_M(n-1)X_M^*(n) \\ &- K_M(n)X_M'(n)P_M(n-1)D_M(n-1) \\ &- \frac{1}{w}d(n)K_M(n)X_M'(n)P_M(n-1)X_M^*(n) \\ &= h_M(n-1) + K_M(n)[d(n) - X_M'(n)h_M(n-1)] \end{aligned} \tag{2.1.67}$$

Ta thấy rằng $X_M'(n)h_M(n-1)$ là đầu ra của bộ lọc thích nghi ở thời điểm n dựa vào hệ số bộ lọc ở thời điểm (n-1). Do đó

$$X_M'(n)h_M(n-1) = d'(n,n-1) \equiv d'(n) \tag{2.1.68}$$

Và

$$e_M(n,n-1) = d(n) - d'(n,n-1) \equiv e_M(n) \tag{2.1.69}$$

$$h_M(n) = h_M(n-1) + K_M(n)e_M(n) \tag{2.1.70}$$

Tương đương

$$h_M(n) = h_M(n-1) + P_M(n)X_M^*(n)e_M(n) \tag{2.1.71}$$

Giả sử ta có hệ số bộ lọc tối ưu $h_M(n-1)$, ma trận $P_M(n-1)$ và vector $X_M(n-1)$. Khi nhận được một tín hiệu mới ta lập vector $X_M(n)$ bằng cách tách phần $x(n-1)$ từ $X_M(n-1)$ và cộng thêm phần $x(n)$. Và hệ số bộ lọc được tính một cách đệ quy như sau:

1. Tính đầu ra của bộ lọc:

$$d'(n) = X'_M(n)h_M(n-1)$$

(2.1.72)

2. Tính lỗi

$$e_M(n) = d(n) - d'(n)$$

(2.1.73)

3. Tính vector Kalman

$$K_M(n) = \frac{1}{w + X'_M(n)P_M(n-1)X^*_M(n)} P_M(n-1)X^*_M(n)$$

(2.1.74)

4. Cập nhật ma trận đảo của ma trận tương quan

$$P_M(n) = \frac{1}{w}[P_M(n-1) - K_M(n)X'_M(n)P_M(n-1)]$$

(2.1.75)

5. Cập nhật vector hệ số của bộ lọc

$$h_M(n) = h_M(n-1) + K_M(n)e_M(n)$$

(2.1.76)

Thuật toán đệ quy được thiết lập bởi (2.1.72) qua (2.1.76) gọi là thuật toán bình phương tối thiểu đệ quy (RLS). Ban đầu đặt $h_M(-1) = 0$ và $P_M(-1) = 1/\delta I_M$, δ là số dương nhỏ.

Phần lỗi bình phương trung bình còn dư do việc tối ưu hóa là

$$\xi_{M\,min} = \sum_{l=0}^{n} w^{n-1}|d(l)|^2 - h'_M(n)D^*_M(n)$$

(2.1.77)

Từ (2.1.76) ta thấy các hệ số bộ lọc thay đổi theo thời gian một lượng bằng $K_M(n)e_M(n)$. Do $K_M(n)$ là một vector thứ nguyên, mỗi hệ số bộ lọc sẽ được diều khiển bởi 1 trong những thành phần của $K_M(n)$. Vì vậy, ta có được sự hội tụ nhanh. Ngược lại, biểu thức thay đổi theo thời gian của các hệ số bộ lọc sử dụng trong thuật toán LMS

$$h_M(n) = h_M(n-1) + \Delta X^*_M(n)e_M(n)$$

(2.1.78)

- Tìm thừa số LDU và thuật toán căn bậc hai. Thuật toán LMS chỉ có một thông số Δ để điều khiển tốc độ hội tụ. Thuật toán RLS ở trên rất dễ dàng chấp nhận làm tròn nhiễu trong hoạt động của thuật toán với phép toán độ

chính xác giới hạn. Vấn đề chính của việc làm tròn xảy ra khi cập nhật $P_M(n)$. Để khắc phục vấn đề này, ta có thể khai triển hoặc ma trận tương quan $R_M(n)$ hoặc nghịch đảo của nó $P_M(n)$.

Ta hãy xét khai triển LDU của $P_M(n)$.

$$P_M(n) = L_M(n)\bar{D}_M(n)L_M^H(n) \tag{2.1.79}$$

Với $L_M(n)$ là ma trận (phần dưới) dạng tam giác với các phần tử l_{lk}, $\bar{D}_M(n)$ là đường chéo ma trận và các phần tử δ_k, $L_M^H(n)$ là ma trận (phần trên) tam giác. Các phần tử trên đường chéo của $L_M(n)$ bằng 1. Để thay cho việc tính $P_M(n)$ một cách đệ quy, ta có thể xác định công thức cập nhật $L_M(n)$ và $\bar{D}_M(n)$ một cách trực tiếp.

Từ (2.1.75) và (2.1.79) ta có

$$\begin{aligned} &L_M(n)\bar{D}_M(n)L_M^H(n) \\ &\quad = \frac{1}{w}L_M(n-1) \\ &\quad \times \left[\bar{D}_M(n-1) - \frac{1}{w+\mu_M(n)}V_M(n-1)V_M^H(n-1)\right]D_M^H(n-1) \end{aligned} \tag{2.1.80}$$

Với

$$V_M(n-1) = \bar{D}_M(n-1)L_M^H\left(n-1X_M^*(n)\right) \tag{2.1.81}$$

Phần bên trong ngoặc của (2.1.80) là ma trận Hermitian và có thể được viết dưới dạng

$$\begin{aligned} &\bar{L}_M(n-1)\bar{D}_M(n-1)\bar{L}_M^H(n-1) \\ &\quad = \bar{D}_M(n-1) - \frac{1}{w+\mu_M(n)}L_M(n)\bar{D}_M(n)L_M^H(n)V_M^H(n-1) \end{aligned} \tag{2.1.82}$$

Sau đó thay (2.1.82) vào (2.1.80)

$$\begin{aligned} &L_M(n)\bar{D}_M(n)L_M^H(n) \\ &\quad = \frac{1}{w}L_M(n-1)\bar{L}_M(n-1)\bar{D}_M(n-1)\bar{L}_M^H(n-1)L_M^H(n-1) \end{aligned} \tag{2.1.83}$$

Và

$$L_M(n) = L_M(n-1)\bar{L}_M(n-1)$$
$$\bar{D}_M(n) = \frac{1}{w}\bar{D}_M(n-1)$$

(2.1.84)

Kết quả thuật toán nhận được từ (2.1.84) phụ thuộc trực tiếp vào vector dữ liệu $X_M(n)$ và không phụ thuộc vào bình phương dữ liệu. Vì thế thuật toán bình phương bị loại bỏ và ảnh hưởng của lỗi làm tròn được giảm thiểu.

Thuật toán RLS nhận được từ việc khai triển $R_M(n)$ hoặc $P_M(n)$ được gọi là thuật toán căn bậc hai RLS.

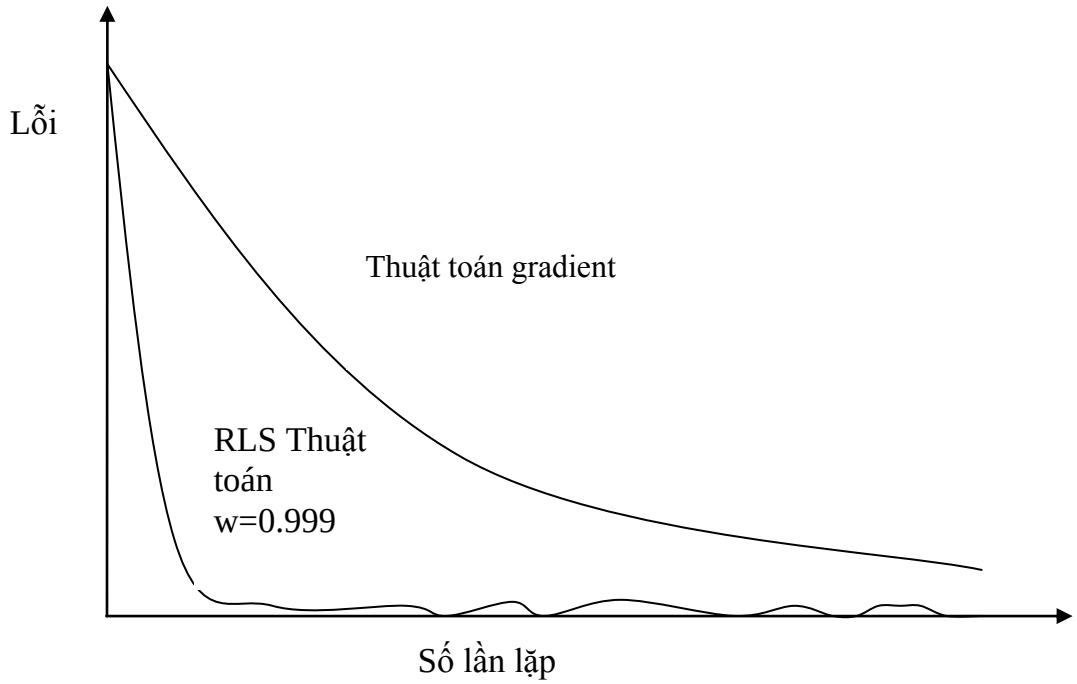
- Thuật toán RLS nhanh

Thuật toán RLS dạng trực tiếp và dạng căn bậc hai có cách tính toán phức tạp tỉ lệ với M^2. Mặt khác, thuật toán giàn RLS (ở 2.3) lại tỉ lệ với M. các thuật toán giàn loại bỏ việc nhân ma trận xuất hiện khi tính $K_M(n)$.

Bằng cách sử dụng các công thức cho giàn LMS ta có thể nhận được các biểu thức theo thời gian của vector khuếch đại Kalman mà hoàn toàn không dùng tới việc nhân ma trận. Các thuật toán phức tạp tỉ lệ với M được gọi là thuật toán RLS nhanh cho bộ lọc FIR dạng trực tiếp.

2.1.5. Các thuộc tính của thuật toán RLS dạng trực tiếp

Thuật toán RLS hơn LMS ở chỗ là có sự hội tụ nhanh. Trạng thái đặc trưng này được thể hiện ở 2.3 (diễn tả tốc độ hội tụ của 2 thuật toán đối với kênh cân bằng FIR thích nghi có độ dài M=11. Ma trận tự tương quan Γ_M dành cho tín hiệu nhận có tỉ lệ giá trị riêng là $\lambda_{max}/\lambda_{min} = 11$. Tất cả các hệ số bộ cân bằng ban đầu được đặt bằng 0. Kích thước bước nhảy thuật toán LMS $\Delta = 0.02$, là giá trị tối ưu đảm bảo cho cả tốc độ hội tụ cả lỗi bình phương trung bình dư.



Hình 2.3 Đồ thị cho thuật toán RLS và LMS

(70 mẫu tín hiệu) trong khi thuật toán LMS không hội tụ trong hơn 600 lần lặp. tốc độ hội tụ này của RLS vô cùng quan trọng trong các ứng dụng khi mà tín hiệu thay đổi nhanh theo thời gian.

Không kể tới chất lượng tự hiệu chỉnh cao, thuật toán RLS của bộ lọc thích ứng FIR có 2 nhược điểm lớn. Một là cách tính toán phức tạp. Thuật toán căn bậc hai tỉ lệ với M^2, RLS nhanh tỉ lệ với M nhưng hệ số tỉ lệ bằng 4 đến 5 lần so với LMS. Nhược điểm thứ hai là đặc tính nhạy của nó khi làm tròn lỗi tích lũy khi tính toán đệ quy. Trong một vài trường hợp, lỗi làm tròn khiến cho thuật toán không ổn định.

Thuộc tính của RLS được nghiên cứu bởi nhiều nhà nghiên cứu. Bảng 2.1 đưa ra kết quả mô phỏng lỗi bình phương trong trạng thái ổn định của thuật toán căn bậc hai RLS, RLS nhanh và LMS với các độ dài từ khác nhau. Việc mô phỏng được thực hiện với bộ cân bằng thích ứng có độ dài M=11. Với chỉ số trọng số lũy thừa đối với RLS là $w = 0.975$ và kích thước bước nhảy $\Delta = 0.025$ đối với LMS. Nhiễu cộng thêm là 0.001, đầu ra MSE với tính toán chính xác là 2.1×10^{-3}.

Ta có thể chỉ ra rằng thuật toán RLS dạng trực tiếp trở nên mất ổn định

và do đó không thể làm việc với các phép toán 16 bits. Đối với thuật toán này, cần 24 bits. Mặt khác, thuật toán căn bậc hai làm việc dưới 9 bits, RLS nhanh làm việc khá tốt dưới 11 bits trong thời gian ngắn, với 500 lần lặp, nếu số lần lặp lớn hơn thuật toán sẽ mất ổn định. Điều đó dẫn gây tích lũy lỗi làm tròn.

Số bit	Thuật toán		
	RLS Square root	Fast RLS	LMS
16	2.17	2.17	2.30
13	2.33	2.21	2.30
11	6.14	3.34	19.0
10	17.6		77.2
9	75.3		311.0
8			1170.0

Bảng 2.1 Độ chính xác của các thuật toán bộ lọc thíc nghi FIR

2.2. Bộ lọc thích nghi dạng thang lưới

Bộ lọc FIR có thể được thực hiện với cấu trúc giàn với thông số giàn được gọi là hệ số phản xạ, được liên kết với các hệ số bộ lọc ở dạng trực tiếp. Có phương pháp để chuyển đổi hệ số bộ lọc FIR thành hệ số phản xạ.

Trong phần này ta nghiên cứu các thuật toán của bộ lọc thích nghi với cấu trúc giàn hoặc giàn hình thang. Các thuật toán này dựa trên phương pháp bình phương tối thiểu và có nhiều thuộc tính mong muốn, đưa ra cách tính toán hiệu quả và lỗi sai số làm tròn được kiểm soát tốt.

2.2.1. Thuật toán thang lưới bình phương tối thiểu hồi qui

Các thuật toán bình phương tối thiểu đệ quy dành cho bộ lọc FIR dạng trực tiếp mô tả trong phần 2.1.4 chỉ đệ quy trong miền thời gian. Độ dài của bộ lọc được cố định. Sự thay đổi độ dài bộ lọc sẽ tạo ra các hệ số bộ lọc mới hoàn toàn khác với các hệ số trước đó.

Ngược lại, bộ lọc giàn là hồi quy bậc, vì thế độ dài một số bộ lọc có thể tăng hoặc giảm mà không ảnh hưởng tới hệ số phản xạ của các bộ lọc còn lại.

Giả sử ta nhận được tín hiệu $x(n-1), l = 1,2,\ldots\ldots n$ và chú ý tới việc ước lượng $x(n)$. Gọi $f_M(l,n)$ là lỗi ước lượng trước đối với việc ước lượng bậc thứ m

$$f_M(l,n) = x(l) + a'_M(n)X_m(l-1) \tag{2.2.1}$$

Với vector $a'_M(n)$ chứa các hệ số ước lượng trước

$$a'_m(n) = [a_m(1,n)a_m(2,n)\ldots.a_m(m,n)] \tag{2.2.2}$$

Và vector dữ liệu $X_M(l-1)$ là

$$X_m(l-1) = [x(l-1)x(l-2)\ldots\ldots x(l-m)] \tag{2.2.3}$$

Các hệ số ước lượng $a_M(n)$ được chọn để tối thiểu hóa lỗi bình phương trọng số thời gian trung bình.

$$\xi_m^f(n) = \sum_{l=0}^{n} w^{n-1}\,|f_m(l,n)|^2 \tag{2.2.4}$$

Việc tối thiểu $\xi f_M(n)$ đồng thời chú ý tới $a_M(n)$ dẫn tới biểu thức tuyến tính

$$R_m(n-1)a_m(n) = -Q_m(n) \tag{2.2.5}$$

Với $R_M(n)$ là ma trận tương quan tín hiệu

$$R_m(n) = \sum_{l=0}^{n} w^{n-1}X_m^*(l)X'_m(l)$$

Và $Q_M(n)$ được định nghĩa là

$$Q_m(n) = \sum_{l=0}^{n} w^{n-1}x(l)X_m^*(l-1) \tag{2.2.6}$$

Từ (2.2.5) ta có

$$a_m(n) = -R_m^{-1}(n-1)Q_m(n) \tag{2.2.7}$$

Giá trị nhỏ nhất của $\xi f_M(n)$ được coi như $Ef_M(n)$

$$E_m^f(n) = \sum_{l=0}^{n} w^{n-1}x^*(l)[x(l) + a'_m(n)X_m(l-1) = q(n) + a'_m(n)Q_m^*(n)] \tag{2.2.8}$$

Với

$$q(n) = \sum_{l=0}^{n} w^{n-1}|x(l)|^2 \tag{2.2.9}$$

(2.2.5) và (2.2.8) có thể ở dạng ma trận

$$\begin{bmatrix} q(n) & Q_m^H(n) \\ Q_m(n) & R_m(n-1) \end{bmatrix}\begin{bmatrix} 1 \\ a_m(n) \end{bmatrix} = \begin{bmatrix} E_m^f(n) \\ O_m \end{bmatrix} \tag{2.2.10}$$

Khi O_m là vector rỗng thứ nguyên m. Cần chú ý rằng

$$\begin{aligned} R_{m+1}(n) &= \sum_{l=0}^{n} w^{n-1}X_{m+1}^*(l)X'_{m+1}(l) \\ &= \sum_{l=0}^{n} w^{n-1}\begin{bmatrix} x^*(l) \\ X_m^*(l-1) \end{bmatrix}[x(l)X'_m(l-1)] \\ &= \begin{bmatrix} q(n) & Q_m^H(n) \\ Q_m(n) & R_m(n-1) \end{bmatrix} \end{aligned} \tag{2.2.11}$$

Ta cũng tối thiểu hóa lỗi bình phương trọng số thời gian trung bình phía sau

$$\xi_m^b(n) = \sum_{l=0}^{n} w^{n-1}[g_m(l,n)]^2 \tag{2.2.12}$$

Với lỗi phía sau là

$$g_m(l,n) = x(l-m) + b'_m(n)X_m(l) \tag{2.2.13}$$

Và $b'_m = [b_m(1,n)b_m(2,n) \dots \dots b_m(m,n)]$ là vector hệ số ước lượng lùi. Việc tối thiểu hóa $\xi_m^b(n)$ dẫn tới biểu thức

$$R_m(n)b_m(n) = -V_m(n) \tag{2.2.14}$$

$$b_m(n) = -R_m^{-1}(n)V_m(n) \tag{2.2.15}$$

Với

$$V_m(n) = \sum_{l=0}^{n} w^{n-1}x(l-m)X_m^*(l) \tag{2.2.16}$$

Giá trị nhỏ nhất của $\xi_m^b(n)$ là $E_m^b(n)$

$$E_m^b(n) = \sum_{l=0}^{n} w^{n-1}[x(l-m) + b_m'(n)X_m(l)]x^*(l-m)$$
$$= v(n) + b_m'(n)V_m^*(n) \tag{2.2.17}$$

Với $v(n)$ là đại lượng vô hướng.

$$v(n) = \sum_{l=0}^{n} w^{n-1}|x(l-m)|^2 \tag{2.2.18}$$

Từ (2.2.14) và (2.2.17) có

$$\begin{bmatrix} R_m(n) & V_m(n) \\ V_m^H(n) & v(n) \end{bmatrix}\begin{bmatrix} b_m(n) \\ 1 \end{bmatrix} = \begin{bmatrix} O_m \\ E_m^b(n) \end{bmatrix} \tag{2.2.19}$$

Ma trận tự tương quan cũng được ước lượng

$$R_{m+1}(n) = \sum_{l=0}^{n} w^{n-1}\begin{bmatrix} X_m^*(l) \\ x_m^*(l-m) \end{bmatrix}X_m'(l)x(l-m) = \begin{bmatrix} R_m(n) & V_m(n) \\ V_m^H(n) & v(n) \end{bmatrix} \tag{2.2.20}$$

Như vậy ta đã có công thức ước lượng bình phương tối thiểu tiến và lùi của bậc m.

Tiếp theo ta sẽ suy ra các biểu thức bậc khác. Ta dùng hai ma trận đảo đồng nhất của ma trận dạng

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \tag{2.2.21}$$

$$A^{-1} = \begin{bmatrix} A_{11}^{-1} + A_{11}^{-1}A_{12}\bar{A}_{22}^{-1}A_{21}A_{11}^{-1} & -A_{11}^{-1}A_{12}\bar{A}_{22}^{-1} \\ -\bar{A}_{22}^{-1}A_{21}A_{11}^{-1} & \bar{A}_{22}^{-1} \end{bmatrix} \tag{2.2.22}$$

Và

$$A^{-1} = \begin{bmatrix} \bar{A}_{11}^{-1} & -\bar{A}_{11}^{-1}A_{12}A_{22}^{-1} \\ -A_{22}^{-1}A_{21}\bar{A}_{11}^{-1} & A_{22}^{-1}A_{21}\bar{A}_{11}^{-1}A_{12}A_{22}^{-1} + A_{22}^{-1} \end{bmatrix} \tag{2.2.23}$$

Với

$$\bar{A}_{11} = A_{11} - A_{21}A_{12}A_{22}^{-1}$$
$$\bar{A}_{22} = A_{22} - A_{21}A_{12}A_{11}^{-1} \tag{2.2.24}$$

- Hồi quy nâng bậc

Dùng biểu thức (2.1.22) để tìm ma trận đảo của $R_{m+1}(n)$. Đầu tiên, ta có

$$\begin{aligned} \bar{A}_{22} &= v(n) - V_m^H(n)R_m^{-1}(n)V_m(n) \\ &= v(n) + b_m'(n)V_m^*(n) = E_m^b(n) \end{aligned} \tag{2.2.25}$$

Và

$$A_{12}A_{11}^{-1} = R_m^{-1}(n)V_m(n) = -b_m(n) \tag{2.2.26}$$

Do vậy

$$R_{m+1}^{-1}(n) \equiv P_{m+1}(n) = \begin{bmatrix} P_m(n) + \dfrac{b_m(n)b_m^H(n)}{E_m^b(n)} & \dfrac{b_m(n)}{E_m^b(n)} \\ \dfrac{b_m^H(n)}{E_m^b(n)} & \dfrac{1}{E_m^b(n)} \end{bmatrix} \tag{2.2.27}$$

Tương đương với

$$P_{m+1}(n) = \begin{bmatrix} P_m(n) & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{E_m^b(n)} \begin{bmatrix} b_m(n) \\ 1 \end{bmatrix} [b_m^H(n) \quad 1]$$

Thay n bằng (n-1) vào (2.2.27) và nhân sau kết quả với $-Q_m(n)$ ta có

$$a_{m+1}(n) = -P_{m+1}(n-1)Q_{m+1}(n)$$
$$= \begin{bmatrix} P_m(n-1) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -Q_m(n) \\ \dots \end{bmatrix}$$
$$- \frac{1}{E_m^b(n-1)} \begin{bmatrix} b_m(n-1) \\ 1 \end{bmatrix} [b_m^H(n-1) \quad 1] Q_{m+1}(n)$$
$$= \begin{bmatrix} a_m(n) \\ 0 \end{bmatrix} - \frac{k_{m+1}(n)}{E_m^b(n-1)} \begin{bmatrix} b_m(n-1) \\ 1 \end{bmatrix}$$

(2.2.28)

Với $k_{m+1}(n)$ là đại lượng vô hướng

$$k_{m+1}(n) = [b_m^H(n-1) \quad 1] Q_{m+1}(n)$$

(2.2.29)

(2.1.28) là sự đệ quy dạng Levinson cho các hệ số bộ lọc ước lượng.

Tương tự để nhận được biểu thức cập nhật cho $b_m(n)$ ta dùng ma trận đảo của $R_{m+1}(n)$. Trong trường hợp này ta có

$$\bar{A}_{11} = q(n) - Q_m^H(n) R_m^{-1}(n-1) Q_m(n) = q(n) + a_m' Q_m^*(n) = E_m^f(n)$$

(2.2.30)

Và

$$A_{21} A_{22}^{-1} = R_m^{-1}(n-1) Q_m(n) = -a_m(n)$$

(2.2.31)

Do đó

$$P_{m+1}(n) = \begin{bmatrix} \frac{1}{E_m^f(n)} & \frac{a_m^H(n)}{E_m^f(n)} \\ \frac{a_m(n)}{E_m^f(n)} & P_m(n-1) + \frac{a_m(n) a_m^H(n)}{E_m^f(n)} \end{bmatrix}$$

Hoặc tương đương

$$P_{m+1}(n) = \begin{bmatrix} 0 & 0 \\ 0 & P_m(n-1) \end{bmatrix} + \frac{1}{E_m^f(n)} \begin{bmatrix} 1 \\ a_m(n) \end{bmatrix} [1 \quad a_m^H(n)]$$

(2.2.32)

Nếu nhân sau (2.2.32) bởi $-V_{m+1}(n)$ ta có

$$b_{m+1}(n) = \begin{bmatrix} 0 & 0 \\ 0 & P_m(n-1) \end{bmatrix} \begin{bmatrix} \cdots \\ -V_{m+1}(n) \end{bmatrix} - \frac{1}{E_m^f(n)} \begin{bmatrix} 1 \\ a_m(n) \end{bmatrix} [1 \quad a_m^H(n)] V_{m+1}(n) = \begin{bmatrix} 0 \\ b_m(n-1) \end{bmatrix} - \frac{k_{m+1}^*(n)}{E_m^f(n)} \begin{bmatrix} 1 \\ a_m(n) \end{bmatrix}$$

(2.2.33)

Khi

$$[1 \quad a_m^H(n)] V_{m+1}(n) = [b_m'(n-1) \quad 1] Q_{m+1}^*(n) = k_{m+1}^*(n)$$

(2.2.34)

Các biểu thức cập nhật cho $Ef_m(n)$ và $E_m^b(n)$ cũng có thể tìm được. Từ định nghĩa của $Ef_m(n)$ ở (2.2.8) ta có

$$E_{m+1}^f(n) = q(n) + a_{m+1}'(n) Q_{m+1}^*(n)$$

(2.2.35)

Thay $a_{m+1}(n)$ ở (2.1.8) vào (2.2.35) ta có

$$E_{m+1}^f(n)(n) = q(n) + \left([a_m(n) \quad 0] \begin{bmatrix} Q_m^*(n) \\ \cdots \end{bmatrix} - \frac{k_{m+1}^*(n)}{E_m^b(n-1)} [b_m'(n-1) \quad 1] Q_{m+1}^*(n) \right) = E_m^f(n) - \frac{|k_{m+1}(n)|^2}{E_m^b(n-1)}$$

(2.2.36)

Tương tự ta có

$$E_{m+1}^b(n) = E_m^b(n-1) - \frac{|k_{m+1}(n)|^2}{E_m^f(n)}$$

(2.2.37)

Bộ lọc giàn đặc trưng bởi cặp biểu thức lỗi tiến và lùi $f_m(n, n-1)$ và $g_m(n, n-1)$. Ta có

$$f_{m+1}(n, n-1) = x(n) + a_{m+1}'(n-1) X_{m+1}(n-1)$$

(2.2.38)

Thay $a'_{m+1}(n-1)$ từ (2.2.28) vào (2.2.38) nhận được

$$\begin{aligned} f_{m+1}(n,n-1) &= x(n) + [a'_m(n-1) \quad 0]\begin{bmatrix} X_m(n-1) \\ \dots \end{bmatrix} \\ &- \frac{k_{m+1}(n-1)}{E_m^b(n-2)}[b'_m(n-2) \quad 1] = X_{m+1}(n-1) \\ &= f_m(n,n-1) - \frac{k_{m+1}(n-1)}{E_m^b(n-2)} \\ &\times [x(n-m-1) + b'_m(n-2)X_m(n-1)] \\ &= f_m(n-1) - \frac{k_{m+1}(n-1)}{E_m^b(n-2)} g_m(n-1,n-2) \end{aligned} \tag{2.2.39}$$

Ta định nghĩa

$$f_m(n) = f_m(n,n-1)$$
$$g_m(n) = g_m(n,n-1) \tag{2.2.40}$$

Vì thế (2.2.39) có thể viết là

$$f_{m+1}(n) = f_m(n) - \frac{k_{m+1}(n-1)}{E_m^b(n-2)} g_m(n-1) \tag{2.2.41}$$

Tương tự

$$g_{m+1}(n,n-1) = x(n-m-1) + b'_{m+1}(n-1)X_{m+1}(n) \tag{2.2.42}$$

Thay $b_{m+1}(n-1)$ từ (2.2.33)

$$g_{m+1}(n,n-1) = g_m(n-1,n-2) - \frac{k^*_{m+1}(n-1)}{E_m^f(n-1)} f_m(n,n-1) \tag{2.2.43}$$

Tương đương

$$g_{m+1}(n) = g_m(n-1) - \frac{k^*_{m+1}(n-1)}{E_m^f(n-1)} f_m(n) \tag{2.2.44}$$

Hai biểu thức đệ quy (2.2.41) và (2.2.44) xác định bộ lọc giàn ở hình 2.3.

Để thuận tiện ta xác định các hệ số phản xạ cho giàn

$$\mathcal{K}_m^f(n) = \frac{-k_m(n)}{E_m^b(n-1)}$$

$$\mathcal{K}_m^b(n) = \frac{-k_m^*(n)}{E_m^{bf}(n-1)}$$

(2.2.45)

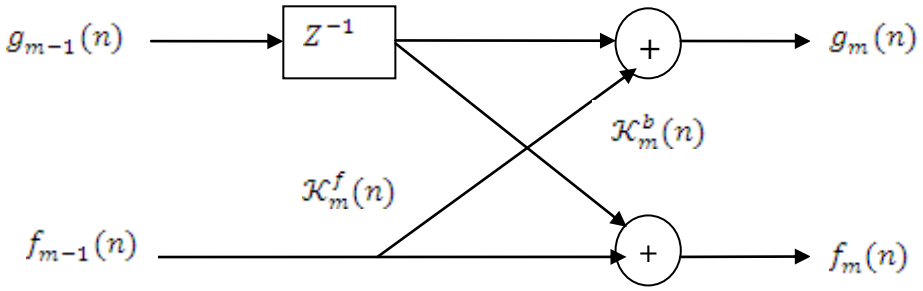
Các điều kiện ban đầu cho việc thay đổi bậc là

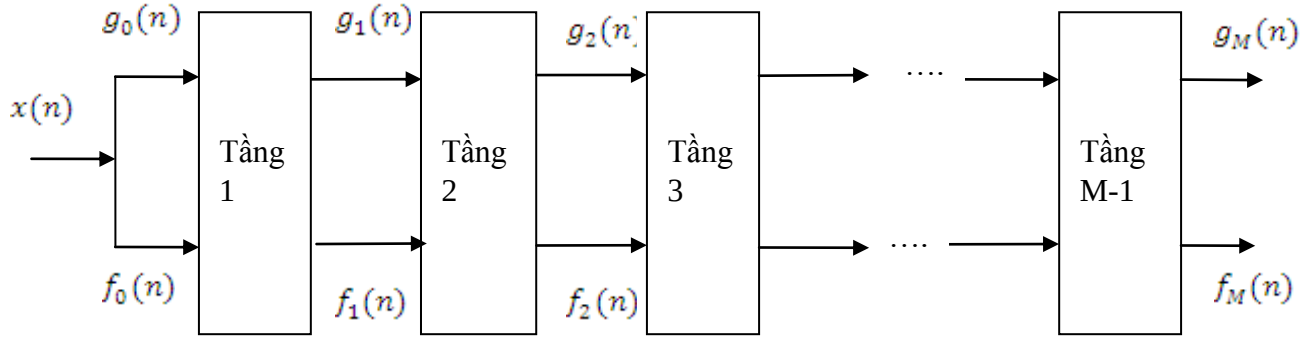
$$f_0(n) = g_0(n) = x(n)$$

$$E_0^f(n) = E_0^b(n) = \sum_{l=0}^{n} w^{n-1}|x(l)|^2 = w\,E_0^f(n-1) + |x(n)|^2$$

(2.2.46)

Và (2.2.46) cũng là biểu thức theo thời gian của $E_0^f(n)$ và $E_0^b(n)$.





Hình 2.4 Bộ lọc lưới bình phương tối thiểu

•Hồi quy điều chỉnh theo thời gian

Ta cần xác định biểu thức theo thời gian $k_m(n)$, nó là cần thiết để bộ lọc giàn trở nên thích nghi và đưa ra các biểu thức theo thời gian cho các hệ số bộ lọc. Ta bắt đầu với

$$k_{m+1}(n) = -V_{m+1}^H(n)\begin{bmatrix}1\\ a_m(n)\end{bmatrix} \tag{2.2.47}$$

Biểu thức theo thời gian cho $V_{m+1}(n)$ là

$$k_{m+1}(n) = wV_{m+1}(n-1) + x(n-m-1)X_{m+1}^*(n) \tag{2.2.48}$$

Biểu thức theo thời gian cho các hệ số ước lượng được xác định như sau. Từ (2.2.6), (2.2.7) và (2.2.8) ta có

$$\begin{aligned} a_m(n) &= -P_m(n-1)Q_m(n)+ \\ &= -\frac{1}{w}[P_m(n-2) - K_m(n-1)X_m'(n-1)P_m(n-2)] \\ &\times [wQ_m(n-1) + x(n)X_m^*(n-1)] \\ &= a_m(n-1) - K_m(n-1)[x(n) + a_m'(n-1)X_m(n-1)] \end{aligned} \tag{2.2.49}$$

Với $K_m(n-1)$ là vector khuếch đại Kalman ở bước lặp thứ n-1. Nhưng từ (2.2.38) ta có

$$x(n) + a_m'(n-1)X_m(n-1) = f_m(n, n-1) \equiv f_m(n) \tag{2.2.50}$$

Bên cạnh đó, dùng (2.2.15), (2.2.16) và (2.2.63) ta được biểu thức theo thời gian cho các hệ số ước lượng lùi ở dạng

$$b_m(n) = b_m(n-1) - K_m(n)g_m(n) \tag{2.2.51}$$

Từ (2.2.48) và (2.2.50) ta có biểu thức theo thời gian của $k_{m+1}(n)$

$$k_{m+1}(n) = -[wV_{m+1}^H(n-1) + x^*(n-m-l)X'_{m+1}(n)] \times \left(\begin{bmatrix}1\\ a_m(n-1)\end{bmatrix} - \begin{bmatrix}0\\ K_m(n-1)f_m(n)\end{bmatrix}\right)$$
$$= wk_{m+1}(n-1) - wV_{m+1}^H(n-1)\begin{bmatrix}0\\ K_m(n-1)\end{bmatrix}f_m(n) + x^*(n-m-l)X'_{m+1}(n)\begin{bmatrix}1\\ a_m(n-1)\end{bmatrix} - x^*(n-m-l)X'_{m+1}(n)\begin{bmatrix}0\\ K_m(n-1)\end{bmatrix}f_m(n)$$

Nhưng

$$X'_{m+1}(n)\begin{bmatrix}1\\ a_m(n-1)\end{bmatrix} = [x(n)X'_m(n-1)]\begin{bmatrix}1\\ a_m(n-1)\end{bmatrix} = f_m(n) \tag{2.2.53}$$

Và

$$V_{m+1}^H(n-1)\begin{bmatrix}0\\ K_m(n-1)\end{bmatrix} = V_m^H(n-2)K_m(n-1)$$
$$= \frac{V_m^H(n-2)P_m(n-2)X_m^*(n-1)}{w+\mu_m(n-1)} = \frac{-b_m^H(n-2)X_m^*(n-1)}{w+\mu_m(n-1)}$$
$$= -\frac{g_m^*(n-1)x^*(n-m-l)}{w+\mu_m(n-1)} \tag{2.2.54}$$

Với $\mu_m(n-1)$ được định nghĩa ở (2.1.62).

$$X'_{m+1}(n)\begin{bmatrix}0\\ K_m(n-1)\end{bmatrix} = \frac{X'_m(n-1)P_m(n-2)X_m^*(n-1)}{w+\mu_m(n-1)}$$
$$= \frac{\mu_m(n-1)}{w+\mu_m(n-1)} \tag{2.2.55}$$

Thay (2.2.53), (2.2.54) và (2.2.55) vào (2.2.52) ta có

$$k_{m+1}(n) = wk_{m+1}(n-1) + \frac{w}{w+\mu_m(n-1)}f_m(n)g_m^*(n-1) \tag{2.2.56}$$

Ta có một biến mới

$$\alpha_m(n) = \frac{w}{w+\mu_m(n)} \tag{2.2.57}$$

$\alpha_m(n)$ có giá trị thực và $0 < \alpha_m(n) < 1$

(2.2.56) giờ được viết

$$k_{m+1}(n) = wk_{m+1}(n-1) + \alpha_m(n-1)f_m(n)g_m^*(n-1)$$

(2.2.58)

- Điều chỉnh bậc cho $\alpha_m(n)$

Dù $\alpha_m(n)$ có thể được tính trực tiếp với mỗi giá trị của m và n, nhưng tốt hơn hết là dùng biểu thức theo bậc. Đầu tiên, từ định nghĩa $K_m(n)$ ở (2.1.61) ta thấy rằng

$$\alpha_m(n) = 1 - X_m'(n)K_m(n)$$

(2.2.59)

Để có được biểu thức theo thời gian cho $\alpha_m(n)$, cần có biểu thức theo bậc của vector Kalman $K_m(n)$. Nhưng $K_{m+1}(n)$ có thể được viết

$$\begin{aligned} K_{m+1}(n) &= P_{m+1}(n)X_{m+1}^*(n) \\ &= \left(\begin{bmatrix} P_m(n) & 0 \\ 0 & 0 \end{bmatrix}\right) + \frac{1}{E_m^b(n)}\begin{bmatrix} b_m(n) \\ 1 \end{bmatrix}\begin{bmatrix} b_m^H(n) & 1 \end{bmatrix} \times \begin{bmatrix} X_m^*(n) \\ x^*(n-m) \end{bmatrix} \\ &= \begin{bmatrix} K_m(n) \\ 0 \end{bmatrix} + \frac{g_m^*(n,n)}{E_m^b(n)}\begin{bmatrix} b_m(n) \\ 1 \end{bmatrix} \end{aligned}$$

(2.2.60)

Trong đó

$$\begin{aligned} g_m(n,n) &= x(n-m) + b_m'(n)X_m(n) \\ &= x(n-m) + [b_m'(n-1) - K_m'(n)g_m(n)]X_m(n) \\ &= x(n-m) + b_m'(n-1)X_m(n) - g_m(n)K_m'(n)X_m(n) \\ &= g_m(n)[1 - K_m'(n)X_m(n)] \\ &= \alpha_m(n)g_m(n) \end{aligned}$$

(2.2.61)

Do đó biểu thức theo bậc đối với $K_m(n)$ ở (2.2.60) được viết

$$K_{m+1}(n) = \begin{bmatrix} K_m(n) \\ 0 \end{bmatrix} + \frac{\alpha_m(n)g_m^*(n)}{E_m^b(n)}\begin{bmatrix} b_m(n) \\ 1 \end{bmatrix}$$

(2.2.62)

Từ (2.2.62) và (2.1.59) ta có

$$\alpha_{m+1}(n) = 1 - X'_{m+1}(n)K_{m+1}(n)$$
$$= 1 - [X'_m(n)x(n-m)] \times \left(\begin{bmatrix} K_m(n) \\ 0 \end{bmatrix} + \frac{\alpha_m(n)g^*_m(n)}{E^b_m(n)} \begin{bmatrix} b_m(n) \\ 1 \end{bmatrix} \right)$$
$$= \alpha_m(n) - \frac{\alpha_m(n)g^*_m(n)}{E^b_m(n)} [X'_m(n)x(n-m)] \begin{bmatrix} b_m(n) \\ 1 \end{bmatrix}$$
$$= \alpha_m(n) - \frac{\alpha_m(n)g^*_m(n)}{E^b_m(n)} g_m(n,n)$$
$$= \alpha_m(n) - \frac{\alpha^2_m(n)|g_m(n)|^2}{E^b_m(n)}$$

(2.2.63)

Như vậy ta đã nhận được các biểu thức theo bậc và theo thời gian của giàn bình phương tối thiểu như ở hình 2.3. (2.2.41) và (2.2.44) là các biểu thức cơ bản cho lỗi tiến và lùi, thường được gọi là các phần dư. (2.2.36) và (2.2.37) là lỗi bình phương tối thiểu, (2.2.58) là biểu thức theo thời gian cho $k_m(n)$, (2.2.63) là biểu thức theo bậc cho thông số $\alpha_m(n)$. Ta có

$$E^f_m(-1) = E^b_m(-1) = E^b_m(-2) = \epsilon > 0$$
$$f_m(-1) = g_m(-1) = k_m(-1) = 0$$
$$\alpha_m(-1) = 1, \;\; \alpha_{-1}(n) = \alpha_{-1}(n-1) = 1$$

(2.2.64)

- Ước lượng quá trình kết hợp.

Ta cần tìm ước lượng bình phương tối thiểu cho tín hiệu d(n) từ giàn. Giả sử bộ lọc có m+1 hệ số và các hệ số này được xác định để tối thiểu hóa lỗi bình phương trọng số trung bình

$$\xi_{m+1} = \sum_{l=0}^{n} w^{n-1} |e_{m+1}(l,n)|^2$$

(2.2.65)

Khi

$$e_{m+1}(l,n) = d(l) - h'_{m+1}(n)X^*_{m+1}(l)$$

(2.2.66)

Ước lượng tuyến tính

$$d'(l,n) = h'_{m+1}(n)X^*_{m+1}(l)$$

(2.2.67)

Ước lượng này nhận được từ giàn bằng cách dùng phần dư $g_m(n)$, và được gọi là ước lượng quá trình kết hợp. Ở (2.1.4) ta đã có các hệ số của bộ lọc thích nghi được tối thiểu hóa bởi biểu thức

$$h_{m+1}(n) = P_{m+1}(n)D_{m+1}(n) \tag{2.2.68}$$

Ta có $h_m(n)$ thỏa mãn biểu thức theo thời gian ở (2.1.76)

Từ (2.2.68) và (2.2.27) ta có

$$h_{m+1}(n) = \begin{bmatrix} P_m(n) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} D_m(n) \\ \dots \end{bmatrix} + \frac{1}{E_m^b(n)} \begin{bmatrix} b_m(n) \\ 1 \end{bmatrix} [b_m^H(n) \quad 1] D_{m+1}(n) \tag{2.2.69}$$

Đại lượng vô hướng

$$\delta_m(n) = [b_m^H(n) \quad 1] D_{m+1}(n) \tag{2.2.70}$$

(2.2.69) có thể viết

$$h_{m+1}(n) = \begin{bmatrix} h_m(n) \\ 0 \end{bmatrix} + \frac{\delta_m(n)}{E_m^b(n)} \begin{bmatrix} b_m(n) \\ 1 \end{bmatrix} \tag{2.2.71}$$

$\delta_m(n)$ thỏa mãn biểu thức theo thời gian, biểu thức này có được từ biểu thức theo thời gian của $b_m(n)$ và $D_m(n)$ ở (2.2.51) và (2.2. 66).

$$\begin{aligned} \delta_m(n) = [b_m^H(n-1) - K_m^H(n)g_m^*(n) \quad 1][wD_{m+1}(n-1) + d(n)X_{m+1}^*(n)] \\ = w\delta_m(n-1) + [b_m^H(n-1) \quad 1]d(n)X_{m+1}^*(n) \\ - wg_m^*(n)[K_m^H(n) \quad 0]D_{m+1}(n-1) \\ - g_m^*(n)d(n)[K_m^H(n) \ \ 0]X_{m+1}^*(n) \end{aligned} \tag{2.2.72}$$

Nhưng

$$[b_m^H(n-1) \quad 1]X_{m+1}^*(n) = x^*(n-m) + b_m^H(n-1)X_m^*(n) = g_m^*(n) \tag{2.2.73}$$

Và

$$[K_m^H(n) \quad 0]D_{m+1}(n-1)$$
$$= \frac{1}{w+\mu_m(n)}[X'_m(n)P_m(n-1)X_m^*(n) \quad 0]\begin{bmatrix} D_m(n-1) \\ \dots \end{bmatrix}$$
$$= \frac{1}{w+\mu_m(n)}X'_m(n)h_m(n-1)$$

(2.2.74)

$$[K_m^H(n) \quad 0]\begin{bmatrix} X_m^*(n) \\ \dots \end{bmatrix} = \frac{1}{w+\mu_m(n)}X'_m(n)P_m(n-1)X_m^*(n) = \frac{\mu_m(n)}{w+\mu_m(n)}$$

(2.2.75)

Thay kết quả ở (2.2.73) qua (2.2.75) vào (2.2.72) ta có

$$\delta_m(n) \equiv w\delta_m(n-1) + \alpha_m(n)g_m^*(n)e_m(n)$$

(2.2.76)

Như vậy ta đã có biểu thức theo bậc cho $\alpha_m(n)$ và $g_m(n)$. Với $e_0(n) = d(n)$, biểu thức theo bậc cho $e_m(n)$ là

$$e_m(n) = e_m(n, n-1) = d(n) - h'_m(n-1)X_m(n)$$
$$= d(n) - [h'_m(n-1) \quad 0]\begin{bmatrix} X_m(n-1) \\ \dots \end{bmatrix}$$
$$- \frac{\delta_{m-1}(n-1)}{E_{m-1}^b(n-1)}[b'_m(n-1) \quad 1]X_m(n)$$
$$= e_m(n-1) - \frac{\delta_{m-1}(n-1)g_m(n-1)}{E_{m-1}^b(n-1)}$$

(2.2.77)

Ước lượng đầu ra d(n) của giàn bình phương tối thiểu là

$$d'(n) = h'_{m+1}(n-1)X_{m+1}(n)$$

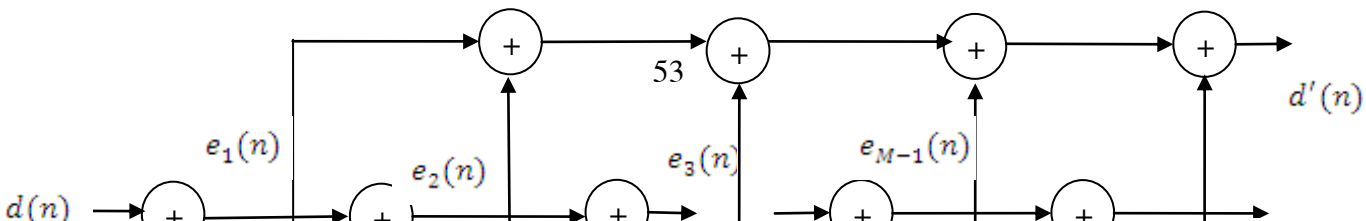
(2.2.78)

Nhưng $h'_{m+1}(n-1)$ không được tính toán cụ thể. Bằng cách dùng $h_{m+1}(n)$ như ở (2.2.71) ta có

$$d'(n) = \sum_{k=0}^{M-1} \frac{\delta_k(n-1)}{E_k^b(n-1)}g_k(n)$$

(2.2.79)

Nói cách khác, ước lượng đầu ra là tổng trọng số tuyến tính của phần dư tiến $g_k(n)$.



Hình 2.5 Bộ lọc thang lưới thích nghi RLS

Dạng ưu tiên của thuật toán giàn hình thang RLS được phân biệt với dạng khác của thuật toán, gọi là dạng lùi. Ở dạng lùi này, $h_m(n-1)$ được thay bởi $h_m(n)$ để ước lượng d(n). Trong bộ cân bằng kênh hoặc khử phản hồi, dạng lùi không sử dụng được vì không thể tính toán $h_m(n)$ để tính d(n).

- Thuật toán thang lưới RLS cải tiến.

Có thể xây dựng các biểu thức khác mà không ảnh hưởng tới tính tối ưu của thuật toán. Tuy nhiên, một vài biến thể của thuật toán vượt hẳn về số lượng khi phép toán điểm ấn định được dùng trong thuật toán.

Đầu tiên, ta có mối liên hệ giữa phần lỗi dư tiến và lùi

Lỗi tiến

$$f_m(n,n-1) \equiv f_m(n) = x(n) + a'_m(n-1)X_m(n-1)$$
$$g_m(n,n-1) \equiv g_m(n) = x(n-m) + b'_m(n-1)X_m(n) \quad (2.2.80)$$

Lỗi lùi

$$f_m(n,n) = x(n) + a'_m(n)X_m(n-1)$$
$$g_m(n,n) = x(n-m) + b'_m(n)X_m(n)$$

(2.2.81)

hệ thức giữa (2.2.80) và (2.2.81) là

$$f_m(n,n) = \alpha_m(n-1)f_m(n)$$
$$g_m(n,n) = \alpha_m(n)g_m(n)$$

(2.2.82)

Thứ hai, ta tìm được các biểu thức theo thời gian cho lỗi bình phương tối thiểu tiến và lùi. Từ (2.2.8) và (2.2.50) ta có

$$\begin{aligned} E_m^f(n) &= q(n) + a'_m(n)Q_m^*(n) \\ &= q(n) + [a'_m(n-1) - K'_m(n-1)f_m(n)] \\ &\times [wQ_m^*(n-1) + x^*(n)X_m(n-1)] \\ &= wE_m^f(n-1) + \alpha_m(n-1)|f_m(n)|^2 \end{aligned}$$

(2.2.83)

Tương tự từ (2.2.17) và (2.2.51) ta có

$$E_m^b(n) = wE_m^b(n-1) + \alpha_m(n)|g_m(n)|^2$$

(2.2.84)

Thứ ba, ta có biểu thức theo thời gian cho vector hệ số khuếch đại Kalman, vector này không được dùng trong thuật toán giàn mà sử dụng trong các thuật toán nhanh. Để có được biểu thức này, ta cũng dùng các biểu thức theo thời gian của hệ số ước lượng tiến và lùi cho bởi (2.2.50) và (2.2.51). Ta có

$$\begin{aligned} K_m(n) &= P_m(n)X_m^*(n) \\ &= \begin{bmatrix} 0 & 0 \\ 0 & P_{m-1}(n-1) \end{bmatrix} \begin{bmatrix} x^*(n) \\ X_{m-1}^*(n-1) \end{bmatrix} \\ &+ \frac{1}{E_{m-1}^f(n)} \begin{bmatrix} 1 \\ a_m(n-1) \end{bmatrix} [1 \quad a_{m-1}^H(n)] \begin{bmatrix} x^*(n) \\ X_{m-1}^*(n-1) \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ K_{m-1}(n-1) \end{bmatrix} + \frac{f_{m-1}^*(n,n)}{E_{m-1}^f(n)} \begin{bmatrix} 1 \\ a_m(n-1) \end{bmatrix} \equiv \begin{bmatrix} C_{m-1}(n) \\ c_{mm}(n) \end{bmatrix} \end{aligned}$$

(2.2.85)

Khi đặt $C_{m-1}(n)$ chứa thành phần đầu tiên của $K_m(n)$ và $C_{mm}(n)$ chứa các thành phần cuối cùng. Từ (2.2.60) ta có biểu thức theo bậc cho $K_m(n)$ là

$$K_m(n) = \begin{bmatrix} K_{m-1}(n) \\ 0 \end{bmatrix} + \frac{g_{m-1}^*(n,n)}{E_{m-1}^b(n)} \begin{bmatrix} b_{m-1}(n) \\ 1 \end{bmatrix}$$

(2.2.86)

Cho (2.2.85) và (2.2.86) bằng nhau ta có

$$c_{mm}(n) = \frac{g^*_{m-1}(n,n)}{E^b_{m-1}(n)}$$

(2.2.87)

Và do đó

$$K_{m-1}(n) + c_{mm}(n)b_{m-1}(n) = C_{m-1}(n)$$

(2.2.88)

Thay (2.2.51) vào (2.2.88) ta có biểu thức theo thời gian cho vector hệ số khuếch đại Kalman

$$K_{m-1}(n) = \frac{C_{m-1}(n) - c_{mm}(n)b_{m-1}(n-1)}{1 - c_{mm}(n)g_{m-1}(n)}$$

(2.2.89)

Đó cũng là biểu thức theo thời gian cho đại lượng vô hướng $\alpha_m(n)$. Từ (2.2.63) ta có

$$\begin{aligned}\alpha_m(n) &= \alpha_{m-1}(n) - \frac{\alpha^2_{m-1}(n)|g_{m-1}(n)|^2}{E^b_{m-1}(n)} \\ &= \alpha_{m-1}(n)[1 - c_{mm}(n)g_{m-1}(n)]\end{aligned}$$

(2.2.90)

Từ (2.2.85) ta có

$$\alpha_m(n) = 1 - X'_m(n)K_m(n) = \alpha_{m-1}(n-1)\left[1 - \frac{f^*_{m-1}(n,n)f_{m-1}(n)}{E^f_{m-1}(n)}\right]$$

(2.2.91)

Cho (2.2.90) và (2.2.91) bằng nhau ta có biểu thức theo thời gian cho $\alpha_m(n)$ là

$$\alpha_{m-1}(n) = \alpha_{m-1}(n-1)\left[\frac{1 - \frac{f^*_{m-1}(n,n)f_{m-1}(n)}{E^f_{m-1}(n)}}{1 - c_{mm}(n)g_{m-1}(n)}\right]$$

Cuối cùng, ta cần phân biệt giữa những phương pháp khác nhau để cập nhật các hệ số phản xạ cho bộ lọc giàn và hình thang là phương pháp trực tiếp và gián tiếp.

Trong phương pháp gián tiếp

$$\mathcal{K}_{m+1}^{f}(n) = -\frac{k_{m+1}(n)}{E_m^b(n-1)} \tag{2.2.93}$$

$$\mathcal{K}_{m+1}^{b}(n) = \frac{k_{m+1}^{*}(n)}{E_m^f(n)} \tag{2.2.94}$$

$$\xi_m(n) = -\frac{\delta_m(n)}{E_m^b(n)} \tag{2.2.95}$$

Với $k_{m+1}(n)$ là theo như (2.2.58), $\delta_m(n)$ theo như (2.2.76), $E_m^f(n)$ và $E_m^b(n)$ theo như (2.2.83) và (2.2.84). Thay $k_{m+1}(n)$ từ (2.2.58) vào (2.2.93) và dùng (2.2.84), ta có

$$\begin{aligned}\mathcal{K}_{m+1}^{f}(n) &= -\frac{k_{m+1}(n-1)}{E_m^b(n-2)}\left(\frac{wE_m^b(n-2)}{E_m^b(n-1)}\right) - \frac{\alpha_m(n-1)f_m(n)g_m^{*}(n-1)}{E_m^b(n-1)} \\ &= \mathcal{K}_{m+1}^{f}(n-1)\left[1 - \frac{\alpha_m(n-1)|g_m(n-1)|^2}{E_m^b(n-1)}\right] \\ &\quad - \frac{\alpha_m(n-1)f_m(n)g_m^{*}(n-1)}{E_m^b(n-1)} \\ &= \mathcal{K}_{m+1}^{f}(n-1) - \frac{\alpha_m(n-1)f_{m+1}(n)g_m^{*}(n-1)}{E_m^b(n-1)}\end{aligned} \tag{2.2.96}$$

Đây là công thức để cập nhật trực tiếp hệ số phản xạ trong giàn. Tương tự, thay (2.2.58) vào (2.2 94) và dùng (2.2.83) ta có

$$\mathcal{K}_{m+1}^{b}(n) = \mathcal{K}_{m+1}^{b}(n-1) - \frac{\alpha_m(n-1)f^{*}{}_m(n)g_{m+1}(n)}{E_m^f(n)} \tag{2.2.97}$$

Hệ số khuếch đại thang cũng có thể được cập nhật một cách trực tiếp

$$\xi_m(n) = \xi_m(n-1) - \frac{\alpha_m(n)g_m^{*}(n)e_{m+1}(n)}{E_m^b(n)} \tag{2.2.98}$$

Một đặc tính của thuật toán thang lưới là phần dư tiến và lùi sẽ quay trở lại cập nhật các hệ số phản xạ trong tầng giàn và $e_{m+1}(n)$ phản hồi lại để cập nhật $\xi_m(n)$. Vì vậy, thuật toán giàn hình thang được gọi là dạng phản hồi lỗi. Có thể nhận được dạng tương tự dành cho thuật toán giàn hình thang RLS ưu

tiên.

- Thuật toán RLS nhanh.

Có hai phiên bản của RLS nhanh. Thực tế, ta cố định kích thước của giàn và kết hợp ước lượng tiến và lùi ở tầng thứ M-1. Vấn đề còn lại là xác định biểu thức theo thời gian cho vector hệ số khuếch đại Kalman như ở (2.2.89).

$$\bar{K}_M(n) = \frac{1}{w} P_M(n-1) X_M^*(n)$$

Có hai điểm khác biệt cơ bản: thứ nhất, Feast dùng vector hệ số khuếch đại thay thế để thay cho vector hệ số khuếch đại Kalman. Thứ 2, RLS nhanh ước lượng lỗi $g_{M-1}(n)$ qua bộ lọc FIR bằng cách dùng vector hệ số $b_{M-1}(n-1)$, trong khi Feast tính qua cách tính toán vô hướng và chú thích là phần tử cuối cùng của vector hệ số khuếch đại thay thế, $\bar{c}_{MM}(n)$, bằng $-wE_{M-1}^b g_{M-1}(n)$.

Chất lượng của RLS nhanh phụ thuộc nhiều vào việc khởi tạo ban đầu.

- Thuật toán lưới chuẩn hay căn bậc hai.

Một dạng khác của thật toán giàn LS được gọi là căn bậc hai hay thuật toán giàn chuẩn hóa. Để có được thuật toán này, đầu tiên ta định nghĩa góc và độ lớn của lỗi LS thông thường

$$\bar{f}_m(n) = \frac{f_m(n)\sqrt{\alpha_m(n-1)}}{\sqrt{E_m^f(n)}} = \frac{f_m(n,n)}{\sqrt{\alpha_m(n-1)\sqrt{E_m^f(n)}}} \tag{2.2.99}$$

$$\bar{g}_m(n) = \frac{g_m(n)\sqrt{\alpha_m(n)}}{\sqrt{E_m^b(n)}} = \frac{g_m(n,n)}{\sqrt{\alpha_m(n)\sqrt{E_m^b(n)}}} \tag{2.2.100}$$

Kĩ thuật chuẩn hóa góc xuất phát từ thực tế là $\alpha_m(n)$ là cosine của góc giữa khoảng của vector dữ liệu tại (n-l) và n. Thứ hai, ta xác định hệ số phản xạ chuẩn hóa $\rho_m(n)$

$$\rho_m(n) = \frac{k_m(n)}{\sqrt{E^b_{m-1}(n-1)}\sqrt{E^f_{m-1}(n)}} \tag{2.2.101}$$

Dễ dàng để chứng minh rằng cả 3 đại lượng đều có giá trị biên độ nhỏ hơn 1. Do thộc tính này thuật toán giàn LS căn bậc 2 chuẩn hóa là tiện ích đối với việc thực hiện các phép toán điểm ấn định. Thay thế (2.2.99), (2.2.100) và (2.2.101) vào (2.2.58) ta có

$$\begin{aligned}\sqrt{E^b_{m-1}(n-1)}\sqrt{E^f_{m-1}(n)}\,\rho_m(n) \\ = w\sqrt{E^b_{m-1}(n-2)}\sqrt{E^f_{m-1}(n-1)}\,\rho_m(n \\ -1)\sqrt{E^f_{m-1}(n)}\bar{f}_{m-1}(n)\bar{g}_{m-1}(n-1)\end{aligned} \tag{2.2.102}$$

Tương đương

$$\begin{aligned}\rho_m(n) = \sqrt{\frac{wE^f_{m-1}(n-1)}{E^f_{m-1}(n)}}\sqrt{\frac{wE^b_{m-1}(n-2)}{E^b_{m-1}(n-1)}}\,\rho_m(n-1) \\ + \bar{f}_{m-1}(n)\bar{g}_{m-1}(n-1)\end{aligned} \tag{2.2.103}$$

Từ biểu thức theo thời gian của $E^b_m(n)$ và $E^f_m(n)$ ở (2.2.83) và (2.2.84) có thể chỉ ra rằng

$$\frac{wE^f_{m-1}(n-1)}{E^f_{m-1}(n)} = 1 - \frac{\alpha_{m-1}(n-1)|f_{m-1}(n)|^2}{E^f_{m-1}(n)} = 1 - \left|\bar{f}_{m-1}(n)\right|^2 \tag{2.2.104}$$

Và

$$\frac{wE^b_{m-1}(n-1)}{E^b_{m-1}(n)} = 1 - \frac{\alpha_{m-1}(n)|g_{m-1}(n)|^2}{E^b_{m-1}(n)} = 1 - |\bar{g}_{m-1}(n)|^2 \tag{2.2.105}$$

Sau đó thay (2.2.104), (2.2.105) vào (2.2.103) ta có biểu thức theo thời gian cho $\rho_m(n)$.

$$\rho_m(n) = \sqrt{1 - \left|\bar{f}_{m-1}(n)\right|^2}\sqrt{1 - |\bar{g}_{m-1}(n-1)|^2}\,\rho_m(n-1) + \bar{f}_{m-1}(n)\bar{g}_{m-1}(n-1)$$

(2.2.106)

Để có biểu thức theo bậc cho lỗi chuẩn hóa trước và sau, trước tiên ta viết biểu thức theo bậc cho lỗi tiến

$$f_m(n,n) = f_{m-1}(n,n) - \frac{k_m(n)}{E_{m-1}^f(n)} g_{m-1}(n-1,n-1)$$

(2.2.107)

Và

$$g_m(n,n) = g_{m-1}(n-1,n-1) - \frac{k_m(n)}{E_{m-1}^f(n)} f_{m-1}(n,n)$$

(2.2.108)

Sau đó viết (2.2.107) trong điều kiện của lỗi chuẩn hóa ta có

$$\sqrt{\alpha_m(n-1)}\sqrt{E_m^f(n)}\bar{f}_m(n) = \sqrt{\alpha_{m-1}(n-1)}\sqrt{E_{m-1}^f(n)}\bar{f}_{m-1}(n) - \frac{\sqrt{E_{m-1}^f(n)}\sqrt{E_{m-1}^b(n-1)}\,\rho_m(n)}{E_{m-1}^b(n-1)}\sqrt{\alpha_{m-1}(n-1)}\sqrt{E_{m-1}^b(n-1)}\bar{g}_{m-1}(n-1)$$

(2.2.109)

Cuối cùng ta có

$$\begin{aligned}\bar{f}_m(n) &= \sqrt{\frac{\alpha_{m-1}(n-1)}{\alpha_m(n-1)}}\sqrt{\frac{E_{m-1}^f(n)}{E_m^f(n)}}\bar{f}_{m-1}(n) \\ &\quad - \sqrt{\frac{\alpha_{m-1}(n-1)}{\alpha_m(n-1)}}\sqrt{\frac{E_{m-1}^f(n)}{E_m^f(n)}}\rho_m(n)\bar{g}_{m-1}(n-1) \\ &= \sqrt{\frac{\alpha_{m-1}(n-1)}{\alpha_m(n-1)}}\sqrt{\frac{E_{m-1}^f(n)}{E_m^f(n)}}\left[\bar{f}_{m-1}(n) - \rho_m(n)\bar{g}_{m-1}(n-1)\right]\end{aligned}$$

(2.2.110)

Từ biểu thức theo bậc cho $E_m^f(n)$ ở (2.2.36) có thể chỉ ra rằng

$$\frac{E_m^f(n)}{E_{m-1}^f(n)} = 1 - \frac{|k_m(n)|^2}{E_{m-1}^f(n)E_{m-1}^b(n-1)} = 1 - |\rho_m(n)|^2$$

(2.2.111)

Từ (2.2.63) ta có

$$\frac{\alpha_m(n-1)}{\alpha_{m-1}(n-1)} = 1 - \frac{\alpha_{m-1}(n-1)|\bar{g}_{m-1}(n-1)|^2}{E_{m-1}^b(n-1)} = 1 - |\bar{g}_{m-1}(n-1)|^2$$

(2.2.112)

Sau đó bằng cách thay (2.2.111) và (2.2.112) vào (2.2.110) ta có kết quả

$$\bar{f}_m(n) = \frac{\bar{f}_{m-1}(n) - \rho_m(n)\bar{g}_{m-1}(n-1)}{\sqrt{1-|\rho_m(n)|^2}\sqrt{1-|\bar{g}_{m-1}(n-1)|^2}}$$

(2.2.113)

Tương tự ta có biểu thức theo bậc cho lỗi lùi chuẩn hóa là

$$\bar{g}_m(n) = \frac{\bar{g}_{m-1}(n) - \rho_m(n)\bar{f}_{m-1}(n)}{\sqrt{1-|\rho_m(n)|^2}\sqrt{1-\left|\bar{f}_{m-1}(n)\right|^2}}$$

(2.2.114)

Biểu thức (2.2.106), (2.2.113) và (2.2.114) xây dựng nên thuật toán giàn LS chuẩn hóa hay căn bậc 2. Chú ý rằng thuật toán này chỉ chứa 3 biến và chỉ được thực hiện với 3 biểu thức. Do vậy nó có dạng chắc chắn hơn so với các dạng khác của thuật toán giàn LS. Tuy nhiên, nó yêu cầu các phép tính căn bậc 2 các phép tính này mất nhiều thời gian. Vấn đề này thì được giải quyết bằng cách sử dụng bộ xử lí CORDIC, có thể tính căn bậc 2 trong N xung, với N là số bits độ dài từ được tính.

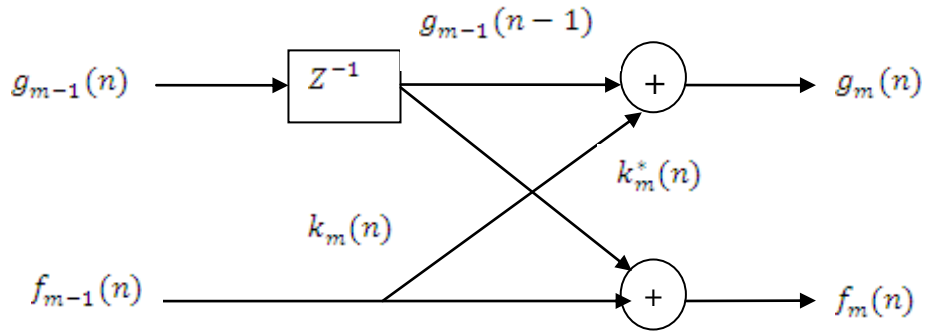
2.2.2. Thuật toán thang lưới Gradient

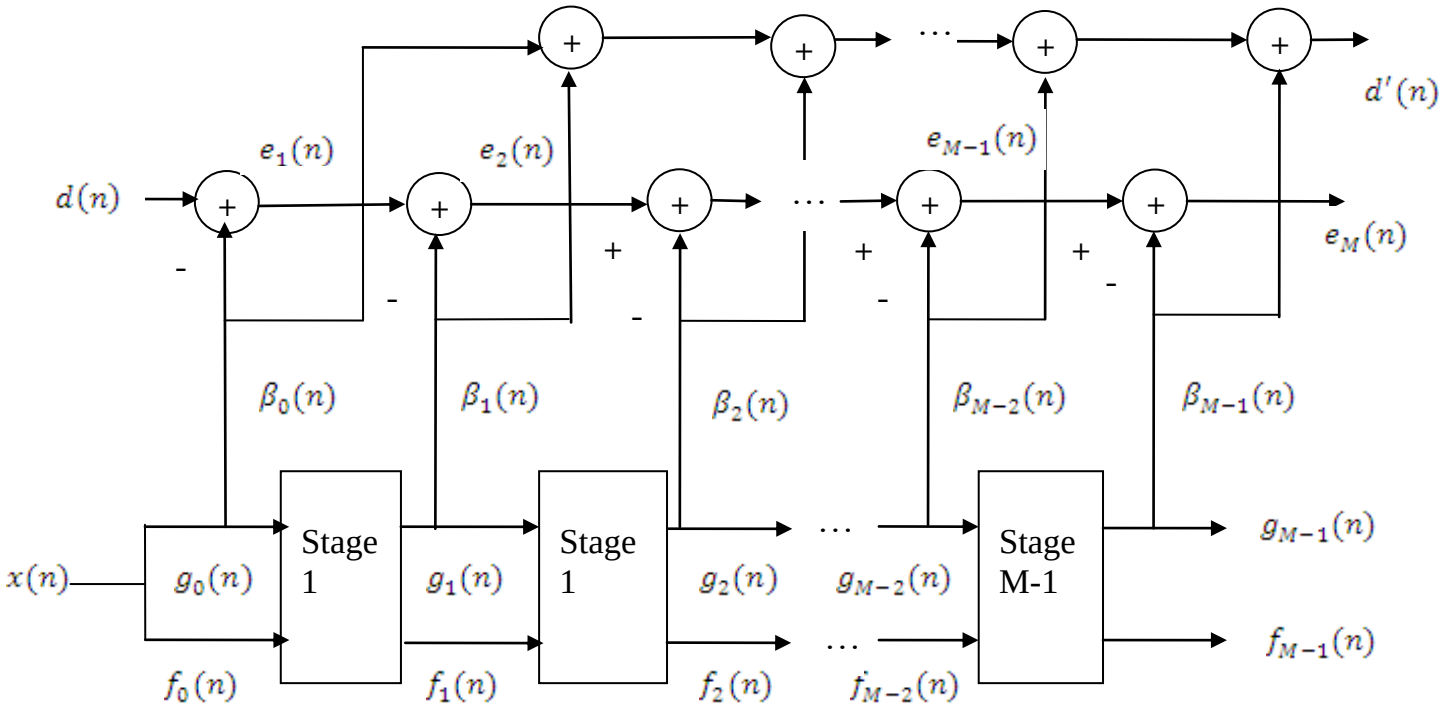
Các thuật toán giàn hình thang được mô tả ở các phần trước thì hoàn phức tạp hơn so với thuật toán LMS nhưng lại có chất lượng tốt hơn. Khi cố gắng đơn giản cách tính toán của dạng này, mà vẫn giữ được tính tối ưu của nó, ta chú ý tới cấu trúc bộ lọc giàn hình thang với số tham số bộ lọc được giảm thiểu. Trong thực tế, cấu trúc bộ lọc thang lưới được thể hiện như ở hình 2.6. Mỗi tầng của giàn đặc trưng bởi hệ thức giữa đầu vào và đầu ra

$$f_m(n) = f_{m-1}(n) - k_m(n)g_{m-1}(n-1)$$
$$g_m(n) = g_{m-1}(n-1) - k_m(n)f_{m-1}(n)$$

(2.2.115)

Với $k_m(n)$ là hệ số phản xạ của tầng thứ m, $f_m(n)$ và $g_m(n)$ là phần dư tiến và lùi.





Hình 2.6 Bộ lọc thang lưới gradient

Dạng bộ lọc này thì giống với dạng được suy ra từ thuật toán Levinson-Durbin, trừ việc $k(n)$ được phép thay đổi theo thời gian. So sánh với thuật toán giàn RLS, thuật toán này hạn chế hơn trong việc ước lượng hệ số tiến và

lùi.

Tham số $k_m(n)$ có thể được tối ưu theo chuẩn MSE hoặc theo phương pháp bình phương tối thiểu. Giả sử ta có chuẩn MSE và chọn các tham số để tối thiểu hóa tổng lỗi bình phương trung bình tiến và lùi.

$$\begin{aligned}\xi_m &= E[|f_m(n)|^2 + |g_m(n)|^2] \\ &= E[|f_{m-1}(n) - k_m(n)g_{m-1}(n-1)|^2 \\ &+ |g_{m-1}(n-1) - k_m(n)f_{m-1}(n)|^2]\end{aligned}$$

Khi ta giảm sự phụ thuộc vào thời gian của k_m

$$k_m = \frac{2E[f_{m-1}(n)g^*_{m-1}(n-1)]}{E[|f_{m-1}(n)|^2] + E[|g_{m-1}(n-1)|^2]} \tag{2.2.116}$$

Chú ý rằng k_m có dạng hệ số tương quan chuẩn hóa.

Khi đặc tính thống kê của tín hiệu chưa được biết đến, ta chấp nhận chuẩn bình phương tối thiểu để xác định $k_m(n)$. Chỉ số đặc trưng được tối thiểu hóa

$$\begin{aligned}\xi_m^{LS} &= \sum_{l=0}^{n} w^{n-1}[|f_m(l)|^2 + |g_m(l)|^2] \\ &= \sum_{l=0}^{n} w^{n-1}[|f_{m-1}(l) - k_m(l)g_{m-1}(l-1)|^2 \\ &+ |g_{m-1}(l-1) - k_m(l)f_{m-1}(l)|^2]\end{aligned} \tag{2.2.117}$$

Ta có

$$k_m(n) = \frac{2\sum_{l=0}^{n} w^{n-1} f_{m-1}(l)g^*_{m-1}(l-1)}{\sum_{l=0}^{n} w^{n-1}[|f_{m-1}(l)|^2 + |g_{m-1}(l-1)|^2]} \tag{2.2.118}$$

Trong (2.2.118), tử và mẫu có thể được viết theo thời gian như sau

$$\begin{aligned}u_m(n) &= wu_m(n-1) + 2f_{m-1}(n)g^*_{m-1}(n-1) \\ v_m(n) &= wv_m(n-1) + |f_{m-1}(l)|^2 + |g_{m-1}(l-1)|^2\end{aligned} \tag{2.2.119}$$

Và

$$k_m(n) = \frac{u_m(n)}{v_m(n)} \tag{2.2.120}$$

Biểu thức theo thời gian của $k_m(n)$

$$k_m(n) = k_m(n-1) + \frac{f_{m-1}(n-1)g_m^*(n-1) + f_m(n-1)g_{m-1}^*(n-2)}{v_m(n-1)}$$

(2.2.121)

Ta định dạng đầu ra của cấu trúc như ở hình 6.4 như một tổ hợp tuyến tính của các phần dư tiến

$$d'(n) = \sum_{k=0}^{M-1} \beta_k(n) g_k(n)$$

(2.2.122)

Với $\beta_k(n)$ là trọng số của phần thang. Giá trị tối ưu của trọng số có thể có được bằng cách tối thiểu MSE giữa tín hiệu mong muốn $d(n)$ và ước lượng của nó. Gọi $e_{m+1}(n)$ là chênh lệch giữa $d(n)$ và ước lượng của nó ở tầng thứ m. Và với $e_m(n) = d(n)$ ta có

$$\begin{aligned} e_{m+1}(n) &= d(n) - \sum_{k=0}^{m} \beta_k(n) g_k(n) \\ &= d(n) - \sum_{k=0}^{m-1} \beta_k(n) g_k(n) - \beta_m(n) g_m(n) \\ &= e_m(n) - \beta_m(n) g_m(n) \end{aligned}$$

(2.2.123)

Lỗi trong (2.2.123) có thể viết ở dạng ma trận

$$e_{m+1}(n) = d(n) - \beta'_{m+1}(n) G_{m+1}(n)$$

(2.2.124)

Với $\beta_{m+1}(n)$ là vector trọng số hình thang và $G_{m+1}(n)$ là vector dư tiến.

Nếu ta giả thiết việc thống kê tín hiệu là không đổi, ta có thể giảm sự phụ thuộc vào thời gian của vector hệ số và chọn $\beta_{m+1}(n)$ thỏa mãn điều kiện trực giao

$$E[e_M(n) G_M^*(n)] = 0$$

(2.2.125)

Thay (2.2.124) vào (2.2.125) và dùng phép toán kì vọng ta có

$$E[d(n) G_M^*(n)] - E[G_M^*(n) G_M'(n)] \beta_M = 0$$

Tương đương

$$\beta_M = \{E[G_M^*(n)G_M'(n)]\}^{-1}E[d(n)G_M^*(n)]$$

(2.2.126)

Một đặc tính quan trọng của phần dư tiến trong bộ lọc giàn được mô tả bởi (2.2.115)

$$E\left[g_k(n)g_j^*(n)\right] = \begin{cases}\xi_k^b, với\ k = j\\ 0, với\ k \neq j\end{cases}$$

(2.2.127)

Do đó ma trận $E[G_M^*(n)G_M'(n)]$ là ma trận chéo và do vậy hệ số khuếch đại thang tối ưu là

$$\beta_m = \frac{1}{\xi_k^b}E[d(n)g_M^*(n)]$$

(2.2.128)

Vấn đề còn lại là điều chỉnh hệ số khuếch đại thang β_m cho thích nghi. Do β_m tối thiểu hóa MSE $d(n)$ và $d'(n)$, lỗi sẽ trực giao với phần dư trước $g_m(n)$ trong $d(n)$. Điều này đưa ra thuật toán gradient ở dạng

$$\beta_m(n+1) = \beta_m(n) + \frac{e_m(n)g_m^*(n)}{\xi_m^b(n)}$$

(2.2.129)

Với $\xi_m^b(n)$ là ước của ξ_m^b và có thể tính toán một cách đệ quy.

$$\xi_m^b(n) = w\xi_m^b(n-1) + |g_m(n)|^2$$

(2.2.130)

Tuy nhiên, việc tính toán như ở (2.2.130) có thể không thực hiện được. Do đó phần dư trước và sau có giá trị bình phương trung bình ban đầu, và biến $v_m(n)$ được ước lượng bằng $2\xi_m^b$. Và (2.2.129) có thể được viết

$$\beta_m(n+1) = \beta_m(n) + \frac{2e_m(n)g_m^*(n)}{v_m(n)}$$

(2.2.131)

Thuật toán ở (2.2.131) dùng để cập nhật hệ số khuếch đại thang là thuật toán gradient, bộ lọc được gọi là bộ lọc giàn hình thang gradient. Thành phần $2/v_m(n)$ đóng vai trò như kích thước bước nhảy.

2.2.3. Thuộc tính của thuật toán thang lưới

- Tốc độ hội tụ

Thuật toán giàn hình thang về cơ bản có tốc độ hội tụ như cấu trúc bội lọc FIR dạng trực tiếp. Tuy thuật toán giàn gradient còn một vài đặc tính của giàn RLS, nhưng lại không tối ưu trong hướng bình phương tối thiểu, do đó nó hội tụ chậm.

- Yêu cầu tính toán

Thuật toán giàn RLS mô tả trong phần trên có cách tính toán phức tạp tỉ lệ thuận với M. Trái lại, thuật toán RLS căn bậc hai tỉ lệ với M^2. Mặt khác, thuật toán nhanh dạng trực tiếp, thuật toán bắt nguồn từ thuật toán giàn RLS, tỉ lệ với M.

- Thuộc tính số

Thuật toán giàn RLS và gradient là lớn về số lượng. Đầu tiên, thuật toán giàn là ổn định về số. Ổn định về số có nghĩa là lỗi ước lượng ở đầu ra có được nhờ tính toán thì bị chặn khi tín hiệu lỗi bị chặn được đưa vào đầu vào. Thứ 2, độ chính xác số của phần tối ưu cũng được so sánh trong mối liên hệ với thuật toán FIR dạng trực tiếp và thuật toán LMS.

Trong thuật toán giàn gradient, các hệ số phản xạ và hệ số khuếch đại thang cũng được cập nhật trực tiếp.

- Sự hoạt động

Cấu trúc bộ lọc giàn có tính kết cấu cao và cho phép tính toán nối truyền liên tiếp. Thực tế, thuật toán RLS và giàn gradient thích ứng với sự hoạt động trong VLSI.

Tổng kết

Trên đây chúng ta đã được thấy các thuật toán thích nghi cho bộ lọc FIR dạng trực tiếp và cấu trúc lưới. Các thuật toán cho bộ lọc FIR dạng trực tiếp là thuật toán LMS đơn giản, thuật toán bình phương tối thiểu đệ quy thời gian(RLS).

Trong đó thuật toán LMS là đơn giản nhất. Nó được sử dụng trong nhiều ứng dụng yêu cầu tốc độ hội tụ chậm. Thuật toán RLS được dùng trong các

ứng dụng yêu cầu tốc độ hội tụ cao hơn.

Các thuật toán dành cho bộ lọc có cấu trúc thang lưới là: thuật toán thang lưới RLS tối ưu, thuật toán thang lưới gradient.

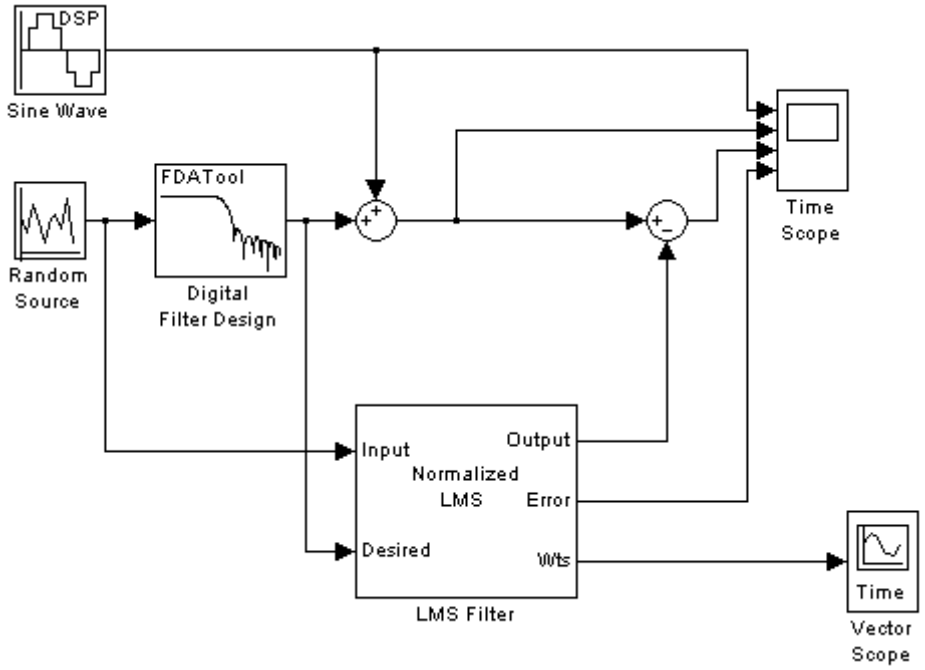
Chương 3.

MÔ PHỎNG ỨNG DỤNG CỦA BỘ LỌC THÍCH NGHI

Bộ lọc thích nghi được sử dụng rộng rãi trong các hệ thống liên lạc, điều khiển. Một số ứng dụng được mô tả trên lí thuyết, một số khác được đưa vào hệ thống anten thích nghi. Trong hệ thống đó, bộ lọc thích nghi được dùng để chỉnh búp sóng. Bộ lọc thích nghi cũng được đưa vào bộ tiếp nhận thông tin số để khử nhiễu liên kí tự và để nhận dạng kênh, đưa vào các kĩ thuật triệt nhiễu để ước lượng và hạn chế thanh phần nhiễu trong tín hiệu.

Trong chương 3, ta đưa mô phỏng về ứng dụng của bộ lọc thích nghi. Bộ lọc thích nghi với thuật toán LMS dùng trong ứng dụng này để ước lượng nhiễu, đồng thời cũng đưa ra hệ số bộ lọc. Hệ số bộ lọc thay đổi theo thời gian và lỗi ước lượng cũng giảm theo sự thay đổi đó.

3.1 Sơ đồ mô phỏng



Hình 3.1 Sơ đồ mô phỏng

Sơ đồ dùng các khối sau:

- Sine wave: cung cấp tín hiệu hình sin ban đầu. tín hiệu ở đầu ra của khối này được đưa trực tiếp vào time scope.
- Random source: tạo nhiễu là 1 tín hiệu bất kì.

- Digital filter design: bộ lọc thông thấp giới hạn tần số của nhiễu.
- Bộ lọc thích nghi sử dụng thuật toán LMS với các thông số bộ lọc được chọn như sau:

+ **Filter length** = 32
+ **Algorithm** = Normalized LMS
+ **Specify step size via** = Dialog
+ **Step size (mu)** = 0.1: xác định chi tiết bước nhảy của bộ lọc
+ **Leakage factor (0 to 1)** = 1.0
+ **Initial value of filter weights** = 0
+ Bỏ chọn **Adapt port**
+ **Reset port** = None
+ Chọn **Output filter weights** check box.

Dựa vào các thông số này, bộ lọc tính toán các trọng số bằng cách sử dụng thuật toán LMS. Do ta chọn **Leakage factor (0 to 1)** = 1.0, giá trị của hệ số bộ lọc phụ thuộc vào các điiều kiện ban đầu của bộ lọc và các giá trị đầu vào trước đó. Giá trị ban đầu của hệ số bộ lọc bằng 0 và sẽ tăng theo thời gian.

- Time scope: dùng đo và thể hiện dạng tín hiệu tạo ra trong quá trình mô phỏng.
- Vector scope: thể hiện các giá trị của hệ số bộ lọc.

3.2 Hoạt động

- Nối đầu ra của **Random source** với đầu vào của bộ lọc LMS.
- Nối đầu ra của **Digital filter design** với đầu **desired** của bộ lọc. Đây là tín hiệu ta muốn bộ lọc sao chép lại.
- Nối đầu ra của ra của bộ lọc với port âm của bộ cộng thứ 2.
- Nối đầu ra của bộ cộng thứ nhất với port dương của bộ cộng thứ 2.

Đầu ra của bộ cộng thứ nhất là tổng của tín hiệu ban đầu và nhiễu tần số thấp, gọi là $s(n) + y$. Đầu ra của bộ lọc là dự đoán tốt nhất của nhiễu tần thấp, gọi là y'. Khi ta trừ 2 tín hiệu này, ta sẽ có gần đúng của tín hiệu ban đầu.

$$s_a = s(n) + y - y'$$

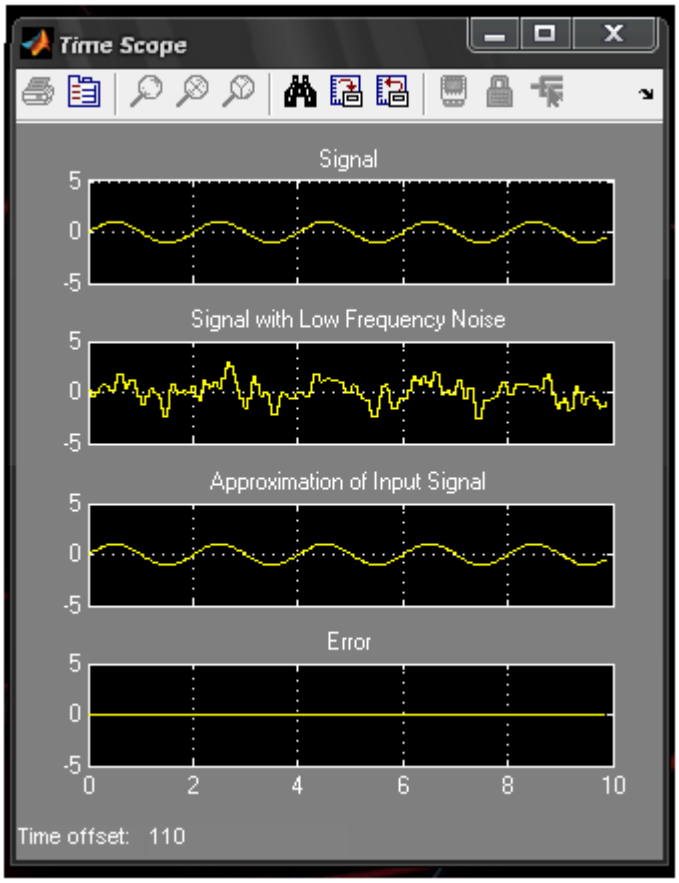
Với $s(n)$ là tín hiệu vào ban đầu.

s_a là tín hiệu gần đúng của $s(n)$.

y là nhiễu được tạo ra từ **Random source** và **Digital filter design.**
y' là gần đúng của y.

Do bộ lọc chỉ có thể tính y' gần đúng với y nên vẫn có sự sai khác giữa tín hiệu ban đầu $s(n)$ và s_a.

- Nối đầu ra của bộ cộng thứ 2 với port thứ 3 của **time scope.**
- Nối đầu ra **erros** của bộ lọc với port thứ 4 của **time scope.**
- Nối đầu ra wts của bộ lọc vơi **vector scope.**



Hình 3.1 Cửa sổ time scope

Kết quả mô phỏng được thể hiện trên cửa sổ của **time scope**. Ta sẽ thấy theo thời gian lỗi sẽ giảm và tín hiệu nhận được sẽ gần giống với tín hiệu ban đầu.

KẾT LUẬN

Trên đây em đã trình bày toàn bộ phần nội dung nghiên cứu vê bộ lọc thích nghi. Để hiểu về hoạt động của bộ lọc, em đã thực hiện tìm hiểu cụ thể về các thuật toán dùng trong đó. Các thuật toán đó được bao gồm các thuật toán cho bộ lọc FIR dạng trực tiếp và cho cấu trúc lưới. Các thuật toán cho bộ lọc FIR dạng trực tiếp là thuật toán LMS đơn giản, thuật toán bình phương tối thiểu đệ quy thời gian(RLS).

Trong đó thuật toán LMS là đơn giản nhất. Nó được sử dụng trong nhiều ứng dụng yêu cầu tốc độ hội tụ chậm. Thuật toán RLS được dùng trong các ứng dụng yêu cầu tốc độ hội tụ cao hơn.Các thuật toán dành cho bộ lọc có cấu trúc thang lưới là: thuật toán thang lưới RLS tối ưu, thuật toán thang lưới gradient. Các thuật toán này giúp bộ lọc thích nghi được với sự thay đổi của tín hiệu đầu vào và tốc độ hội tụ thích hợp. Và từ phần mô phỏng trên, ta cũng thấy rõ hơn về hoạt động của bộ lọc với một thuật toán cụ thể, thuật toán LMS.

Với sự phát triển mạnh mẽ của khoa học kỹ thuật, các ngôn ngữ lập trình mạnh có kèm theo hộp công cụ xử lý số tín hiệu như ngôn ngữ MATLAB thì việc phân tích và thiết kế các bộ lọc số ngày càng trở nên đơn giản (kể cả bộ lọc FIR và bộ lọc IIR) và độ chính xác của phép toán sẽ tăng lên.

Do điều kiện thời gian có hạn cộng với khả năng còn hạn chế nên chắc không tránh khỏi thiếu sót. Vậy rất mong được quý thầy cô chỉ bảo để quyển đồ án này được hoàn thiện.

Em xin chân thành cám ơn thầy Nguyễn Văn Dương, thày giáo hướng dẫn trực tiếp, và các thày cô khác trong bộ môn đã tận tình giúp đỡ và tạo

điều kiện để em hoàn thành quyển đồ án này.

TÀI LIỆU THAM KHẢO

1. John G. Proakis and Dimitris G.Manolakis (1982), *Digital signal processing*,Macmillan Publishing Company.
2. John G. Proakis (1983), *Digital Communication*, McGraw-Hill Book Company.
3. Nguyễn Quốc Trung (2001), *Xử lí tín hiệu và lọc số tập 1 và 2*, Nhà xuất bản khoa học và kĩ thuật.
4. Các tài liệu liên quan khác của
   - Eleftheiou và Falcorner(1987).
   - Cioffi và Kailath(1984).
   - Hsu(1982).