Laboratorul 9

Filtrarea secventelor infinite (lungi) cu filtre FIR.

Se considera ca la intrarea unui filtru de tip FIR se aplica un semnal cu un numar infinit de esantioane. (sau foarte mare comparativ cu numarul de coeficienti ai filtrului) Raspunsului filtrului FIR se obtine folosind produsul de convolutie liniara:

$$y_L[n] = x[n] * h[n] = \sum_{m=0}^{N-1} x[m] \cdot h[n-m] \qquad (1)$$

Dezavantajul principal al filtrelor de tip FIR este ordinul lor mare. De aceea o implementare folosind ecuatii cu diferente este costisitoare din punct de vedere computational. Exista posibilitatea sa se utilizeze convolutia circulara (ceea ce da posibilitatea folosirii algoritmului FFT) pentru calculul convolutiei liniare.

Convolutia circulara a doua semnale x si h care au aceeasi dimensiune N este:

$$y_C[n] = x[n] * h[n] = \sum_{m=0}^{N-1} x[m] \cdot h[< n-m >_N] \qquad (2)$$

Pentru doua semnale (x si h) care au dimensiunea N si respectiv M, convolutia liniara se poate calcula folosind convolutia circulara *prin completare cu zerouri. Se obtin astfel alte doua semnale (notate cu* x' *si* h'*) care au aceeasi dimensiune L=M+N-1.*

De exemplu se considera convolutia liniara dintre $x = [1\ 2\ 3]$ si $h = [4\ 3\ 2\ 1]$. Se completeaza cele doua semnale cu zerouri astfel incat cele doua semnale rezultate sa aiba aceeasi dimensiune 3+4-1=6. Semnalele rezultate sunt $x' = [1\ 2\ 3\ 0\ 0\ 0]$ si $h' = [4\ 3\ 2\ 1\ 0\ 0]$. In figura 1 se prezinta un exemplude calcul a convolutiei liniare si circulare dintre semnalele x si h si respectiv cele completate cu zerouri x' si h'.

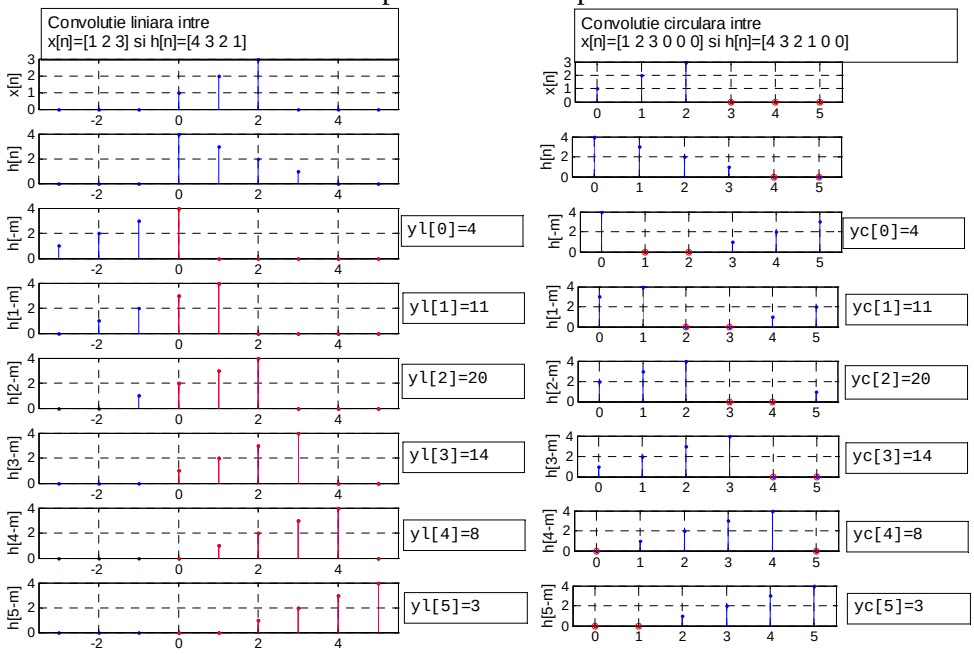

Fig. 1: Exemplu de convolutie liniara dintre [1 2 3], [4 3 2 1] si convolutie circulara dintre [1 2 3 0 0 0], [4 3 2 1 0 0]

Se observa ca zerourile introduse au rolul de a anula esantioanele lui $h[<n-m>_6]$, n=0..5, care in cazul convolutiei circulare nu sunt egale cu cele de la convolutia liniara.

Produsele de convolutie liniara si circulara se mai pot exprima si in forma matriciala (relatiile (4) si (6)) scrise pentru semnalele x[n]=[x[0] x[1] x[2] x[3]] si h[n]=[h[0] h[1]] .

Produsul de convolutie liniara se scrie sub forma:

$$y_L = Hx \tag{3}$$

$$\begin{bmatrix} y_L[0] \\ y_L[1] \\ y_L[2] \\ y_L[3] \\ y_L[4] \end{bmatrix} = \begin{bmatrix} h[0] & 0 & 0 & 0 \\ h[1] & h[0] & 0 & 0 \\ 0 & h[1] & h[0] & 0 \\ 0 & 0 & h[1] & h[0] \\ 0 & 0 & 0 & h[1] \end{bmatrix} \cdot \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix} \tag{4}$$

iar cel de convolutie circulara

$$y_C = H' x' \tag{5}$$

unde $x' = [x[0]\ x[1]\ x[2]\ x[3]\ 0]$ iar H' rezulta prin rotire din $h' = [h[0]\ h[1]\ 0\ 0\ 0]$ sub forma:

$$\begin{bmatrix} y_c[0] \\ y_c[1] \\ y_c[2] \\ y_c[3] \\ y_c[4] \end{bmatrix} = \begin{bmatrix} h[0] & 0 & 0 & 0 & h[1] \\ h[1] & h[0] & 0 & 0 & 0 \\ 0 & h[1] & h[0] & 0 & 0 \\ 0 & 0 & h[1] & h[0] & 0 \\ 0 & 0 & 0 & h[1] & h[0] \end{bmatrix} \cdot \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ 0 \end{bmatrix} \tag{6}$$

Se observa ca $y_L = y_C$.

Folosind proprietatea DFT ($x'[n] \otimes h'[n] = IDFT\{DFT\{x'[n]\} \cdot DFT\{h'[n]\}\}$) se poate calcula mult mai rapid convolutia liniara.

Exista situatii cand trebuie filtrate semnale in timp real. Deoarece nu este practic sa se astepte inregistrarea intregului semnal si apoi sa se calculeze semnalul filtrat, solutia este sa se imparta semnalul x[n] de lungime infinita (sau foarte mare) in semnale de lungime aleasa L.

Rezulta 2 algoritmi care folosesc DFT pentru filtrarea secventelor lungi folosind filtre FIR: overlap-save si overlap-add

Algoritmul overlapp-save

Algoritmul are la baza proprietatea descrisa in relatiile (4) si (6) cu deosebirea ca se completeaza cu zerouri numai semnalul h[n] astfel incat acesta va avea aceeasi dimensiune cu partile componente ale lui x[n].

A1: *Sa se rescrie relatia* (6) *pentru cazul in care x nu se completeaza cu zerouri. Observati diferenta dintre semnalele rezultate* $y_C[n]$ *si* $y_L[n]$.

Folosind proprietatea observata in aplicatia A1 de mai sus, algoritmul overlap-save consta urmatorii pasi:

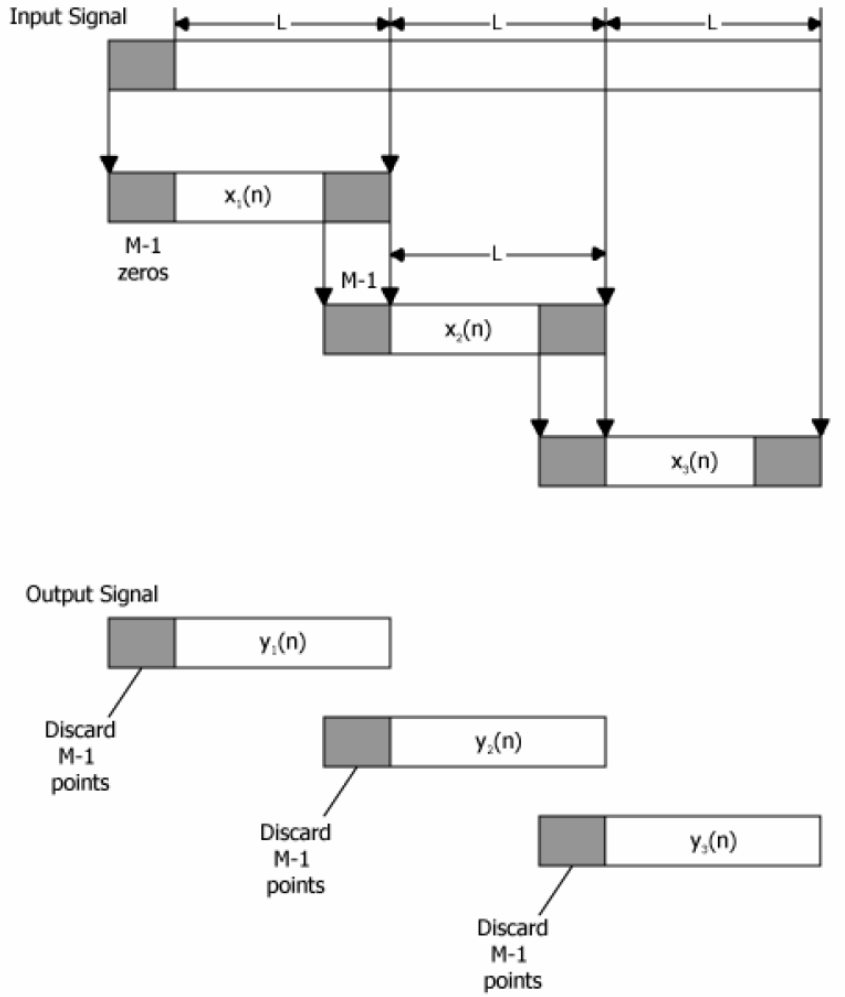

Fig. 2: Algoritmul overlap-save

1. Se *imparte* semnalul x[n] in intervale, fiecare avand dimensiunea L. Fiecare bloc supus filtrarii este format M-1 esantioane ale blocului precedent la care se adauga L esantioane noi; blocurile se noteaza cu $x_k[n]$. Astfel fiecare $x_k[n]$ va fi definit pe intervalul [kL-M+2:(k+1)L], k>1. Initial, semnalul $x_1[n]$ este precedat de M-1 zerouri.
2. Se completeaza cu zerouri *numai* semnalul h[n] astfel incat noul semnal, $h'[n]$, va avea de asemenea dimesiunea L+M-1.
3. Se realizeaza *convolutia circulara* (*folosind DFT*) dintre fiecare $x_k[n]$ si $h'[n]$. Rezulta $y_k[n]$
4. Se *renunta* (se tine cont de observatia aplicatiei A1) la primele M-1 esantioane ale lui $y_k[n]$ (rezulta $y'_k[n]$)
5. Se *concateneaza* semnalele $y'_k[n]$.

A2: *Sa se scrie un script in Matlab care calculeaza raspunsul un filtru FIR h[n]=[3 2 1] la semnalul x[n]=[0:25], n=[0:25]. Se va folosi algoritmul overlap-save si L+M-1=8.*

- *Sa se afiseze semnalele intermediare* $x_k[n]$ *si* $y_k[n]$ *(se va folosi functia Matlab* `axis` *astfel incat reprezentarea pe axa OX sa fie intre 0 si 24).*
- *Verificati corectitudinea algoritmului scris. Se va folosi functia Matlab* `conv` *(returneaza convolutia liniara a doua semnale x si h).*
- *Care este sunt valorile optime ale lui L?*

Algoritmul overlapp-add

Algoritmul overlapp-add are la baza proprietatea de liniaritate a sistemelor liniare si invariante in timp.
In cadrul acestei metode se imparte x[n] in secvente de cate N esantioane. Astfel, semnalul de la intrarea filtrului FIR se scrie:

$$x[n] = \sum_{k=0}^{\infty} x_k[n] \tag{7}$$

unde:

$$x_k[n] = \begin{cases} x[n], & n \in (kL : (k+1)L - 1) \\ 0, & \text{in rest} \end{cases} \tag{8}$$

Substituind $x_k[n]$ in relatia (1) se obtine:

$$y[n] = x[n] * h[n] = \left(\sum_{k=0}^{\infty} x_k[n] \right) * h[n] = \sum_{k=0}^{\infty} \left(x_k[n] * h[n] \right) \tag{9}$$

Deci $y[n]$ se poate calcula ca o suma de produse de convolutie liniara. Fiecare din aceste produse liniare se calculeaza folosind convolutia circulara (se completeaza cu zerouri atat $x_k[n]$ cat si $h[n]$).

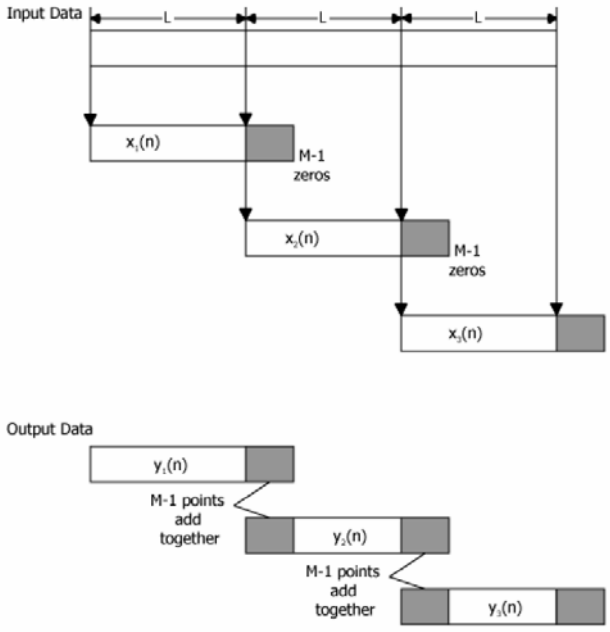

Fig. 3 : Algoritmul overlap-add

Algoritmul overlap-save consta urmatorii pasi:

1. Se imparte semnalul x[n] in semnale $x_k[n]$ de dimensiune L.
2. Se completeaza $x_k[n]$ si h[n] cu zerouri astfel incat ambele semnale sa aiba acelasi dimensiune L+M-1. Rezulta $x'_k[n]$ si respectiv $h'[n]$.
3. Se realizeaza convolutia circulara (*folosind DFT*) dintre fiecare $x'_k[n]$ si $h'[n]$. Rezulta $y'_k[n]$.
4. Se aduna semnalele $y'_k[n]$.

A3: *Sa se scrie un script in Matlab care calculeaza raspunsul un filtru FIR h[n]=[3 2 1] la semnalul x[n]=[0:25], n=[0:25]. Se va folosi algoritmul overlap-add.*

- *Se vor afisa semnalele intermediare $x'_k[n]$ si $y'_k[n]$. Se va folosi functia Matlab* `axis` *astfel incat reprezentarea pe axa OX sa fie intre 0 si 24.*
- *Verificati corectitudinea algoritmului scris. Se va folosi functia Matlab* `conv` *(returneaza convolutia liniara a doua semnale x si h).*
- *Care este sunt valorile optime ale lui L?*
- *Care sunt diferentele dintre algoritmii overlap-save si overlap-add?*