Generarea semnalelor de intrare. Filtrarea datelor

November 2, 2019

1 Aspecte teoretice vizate

Pentru o bună identificare trebuie aplicată o intrare bogată în armonice. Două soluţii clasice sunt semnalul pseudo-aleator binar (SPAB) şi oscilatorul controlat în tensiune (VCO).

O altă problemă ce apare în procesul de identificare este prezenţa zgomotului de măsură. Pentru o mai bună identificare se folosesc algoritmi de filtrare ce sunt aplicaţi atât semnalului de intrare, cât şi semnalului de ieşire.

2 Generarea semnalelor de intrare

2.1 SPAB

Secvenţele pseudo-aleatoare binare sunt succesiuni de impulsuri dreptunghiulare modulate în amplitudine ce aproximează in zgomot alb discret şi au un conţinut bogat de frecvenţe. Acestea se pot genera cu ajutorul unor registre de deplasare cu bucle. Un registru cu N celule generează o secvenţă de dimensiune maxim $2^N - 1$. Principiul de construcţie al unui SPAB este:

1. Se iniţializează toate registrele cu valoarea 1.
2. Se realizează operaţia `XOR` între celulele b_i şi b_j, conform tabelului de mai jos.
3. Se deplasează toate elementele din registru spre dreapta cu o unitate, iar primul element va fi rezultatul operaţiei descrise la punctul 2.

Numărul de celule N	Lungimea secvenţei $L = 2^N - 1$	Biţi în suma $\oplus$ b_i şi b_j
2	3	1 şi 2
3	7	1 şi 3
4	15	3 şi 4
5	31	3 şi 5
6	63	5 şi 6
7	127	4 şi 7
8	255	{2, 3, 4} şi 8
9	511	5 şi 9
10	1023	7 şi 10
11	2047	9 şi 11

2.2 VCO

Voltage Controlled Oscillator – VCO – este un circuit electric ce are ca ieşire un semnal a cărui frecvenţă este dependentă de tensiunea de intrare la un moment dat. În general, se utilizează dependenţa liniară. Relaţia matematică în cazul liniar este:

$$f(t) = f_0 + K_0 \cdot v_{in}(t), \tag{1}$$

unde:

- $f(t)$ este frecvenţa instantanee a oscilatorului la momentul t;
- f_0 este frecvenţa de offset;
- K_0 este senzitivitatea oscilatorului (factorul de amplificare frecvenţial);
- $v_{in}(t)$ tensiunea de intrare.

Implementarea acestuia în MATLAB se realizează prin funcţia `vco`.

3 Filtrarea datelor

3.1 Filtrul median

Filtrul median este o metodă de filtrare bazată pe următorul principiu: pentru fiecare subvector de dimensiune impară $2K + 1$ se consideră valoarea mediană a acestuia. Paşii pentru implementarea filtrului median sunt:

1. se consideră fiecare fereastră formată din $2K + 1$ elemente consecutive;
2. se sortează elementele acestei ferestre şi se reţine valoarea din mijloc (mediana);
3. vectorul final se extinde la capete cu K valori în stânga şi în dreapta folosind prima, respectiv ultima valoare mediană.

O posibilă implementare în MATLAB a paşilor 1 şi 2 este prezentată mai jos:

```
function filtered_signal = median_filt(input_signal,order)

N = length(input_signal);
N1 = floor(order/2);

filtered_signal = zeros(N-2*N1,1);

for i=1+N1:N-N1
    signal_l = input_signal(i-N1:i+N1);
    signal_l = sort(signal_l);
    filtered_signal(i-N1) = signal_l(N1+1);
end

end
```

3.2 Filtrul Tukey53H

Filtrul Tukey53H este o extensie a filtrului median. Acesta se calculează astfel:

1. se aplică o filtrare mediană de lungime 5 semnalului iniţial;
2. se aplică o filtrare mediană de lungime 3 semnalului obţinut la pasul anterior;

3. se aplică o fereastră Hanning semnalului de la pasul 2:

$$u^{(3)}[k] = \frac{1}{4}(u^{(2)}[k-1] + 2 \cdot u^{(2)}[k] + u^{(2)}[k+1]). \tag{2}$$

4. vectorul final se extinde la capete cu câte 4 valori în stânga şi în dreapta folosind prima, respectiv ultima valoare.

O posibilă implementare în MATLAB a unui filtru Tukey53H este prezentată mai jos:

```
function output_signal = tukey53H(input_signal)

N = length(input_signal);

u_median_5 = median_filt(input_signal, 5);
u_median_3 = median_filt(u_median_5, 3);

% apply a Hanning window
output_signal = zeros(N,1);
for i=2:length(u_median_3)-1
    output_signal(i+3) = 1/4*(u_median_3(i-1)+2*u_median_3(i)+u_median_3(i+1))
end
for i=1:4
    output_signal(i) = output_signal(5);
    output_signal(length(output_signal)-i) = output_signal(length(
        output_signal)-5);
end

end
```

3.3 Filtrul LULU

Filtrul LULU este o altă tehnică matematică neliniară de a elimina zgomotele de tip impul dintr-un şir de măsurători. Această tehnică se bazează pe doi operatori idempotenţi: `L` şi `U`.

Un opeartor L de dimensiune K pentru un semnal de intrare discret $u[n]$ se calculează astfel:

1. pentru fiecare fereastră de dimensiune K ce conţine măsurătoarea $u[n]$ se formează secvenţele $seq_1, \ldots, seq_K$;
2. se formează un vector de dimensiune K din minimele fiecărei secvenţe seq_i;
3. ieşirea de pe poziţia n este dată de maximul minimelor considerate anterior;
4. similar cu celelalte metode, primul element şi ultimul element sunt multiplicate pentru a nu reduce dimensiunea semnalului de ieşire.

Un opeartor U de dimensiune K pentru un semnal de intrare discret $u[n]$ se calculează astfel:

1. pentru fiecare fereastră de dimensiune K ce conţine măsurătoarea $u[n]$ se formează secvenţele $seq_1, \ldots, seq_K$;
2. se formează un vector de dimensiune K din maximele fiecărei secvenţe seq_i;
3. ieşirea de pe poziţia n este dată de minimul maximelor considerate anterior;
4. similar cu celelalte metode, primul element şi ultimul element sunt multiplicate pentru a nu reduce dimensiunea semnalului de ieşire.

O aplicare succesivă operatorilor L şi U duce la obţinerea diverselor rezultate. O posibilă implementare în MATLAB este descrisă mai jos:

```
function [y] = LU_op(u, ord, type)

N = length(u);
y = zeros(N,1);
if type == 'L'
    for i= ord:N-ord+1
        u_loc = zeros(1,ord);
        for j = 1:ord
            u_loc(j) = min(u(i-ord+j:i+j-1));
        end
        y(i) = max(u_loc);
    end
    for i=1:ord-1
        y(i) = y(ord);
        y(N-i+1) = y(N-ord);
    end
elseif type == 'U'
    for i= ord:N-ord+1
        u_loc = zeros(1,ord);
        for j = 1:ord
            u_loc(j) = max(u(i-ord+j:i+j-1));
        end
        y(i) = min(u_loc);
    end
    for i=1:ord-1
        y(i) = y(ord);
        y(N-i+1) = y(N-ord);
    end
else
    error('There is no such operator!');
end

end
```

3.4 Alte metode de filtrare

O categorie importantă de filtre de netezire sunt cele de tip medie alunecătoare. Dintre acestea, menţionăm filtru medie alunecătoare de tip Savitzky-Golay, implementat în MATLAB prin funcţia `sgolayfilt`.