CONTRIBUŢII LA PRELUCRAREA NUMERICĂ A SEMNALELOR CU FUNCŢII SPLINE

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea "Politehnica" din Timişoara
în domeniul INGINERIE ELECTRONICĂ
ŞI TELECOMUNICAŢII
de către

Ing. Liliana Stoica

Conducător ştiinţific: prof.univ.dr.ing. Alimpie Ignea
Referenţi ştiinţifici: prof.univ.dr.ing. Teodor Petrescu
prof.univ.dr.ing. Cornelia Gordan
prof.univ.dr.ing. Ioan Naforniţă

Ziua susţinerii tezei: 23.11.2012

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Ingineria Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie Mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor
12. Ingineria sistemelor
13. Inginerie energetică
14. Calculatoare şi tehnologia informaţiei

Universitatea „Politehnica” din Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300159 Timişoara, Bd. Republicii 9,
tel. 0256 403823, fax. 0256 403221
e-mail: editura@edipol.upt.ro

Cuvânt înainte

Prezenta lucrare a fost elaborată pe parcursul programului de doctorat fără frecvenţă susţinut în Departamentul de Măsurări şi Electronică Optică al Facultăţii de Electronică şi Telecomunicaţii din cadrul Universităţii „Politehnica” din Timişoara. Perioada desfăşurării programului doctoral a fost noiembrie 2001 – octombrie 2012.

Teza dezvoltă o cercetare în domeniul interpolării semnalelor. Prelucrarea numerică a semnalelor este utilizată în tot mai multe aplicaţii cotidiene. Aplicaţii diverse din domeniul interpolării imaginilor utilizează metode de interpolare eficiente, prin prisma conceptului de interpolare generalizată. Acestea pot fi utilizate şi în prelucrarea semnalelor unidimensionale. În această teză s-au dezvoltat noi algoritmi de interpolare spline cubică. Unul dintre ei a fost implementat pe un sistem de prelucrare numerică a semnalelor şi permite procesarea datelor în timp real.

Programul de doctorat s-a desfăşurat la început sub conducerea regretatului Prof. dr. ing. Eugen Pop, continuând sub conducerea domnului Prof. dr. ing. Alimpie Ignea. În calitate de conducător de doctorat, domnul Prof. dr. ing. Eugen Pop mi-a oferit cu profesionalism îndrumare deosebită în ceea ce priveşte selectarea literaturii de specialitate adecvată, realizarea referatelor din cadrul doctoratului, scrierea şi publicarea unora dintre lucrările ştiinţifice. Domnul Prof. dr. ing. Alimpie Ignea, cu profesionalism şi răbdare, m-a sprijin în ceea ce priveşte scrierea şi publicarea unora dintre lucrările ştiinţifice şi mai ales pentru finalizarea tezei. Pentru toate acestea le aduc calde mulţumiri.

Sincere mulţumiri aduc membrilor comisiei pentru modul competent de parcurgere a tezei de doctorat şi pentru întocmirea referatelor.

Doresc, de asemenea, să aduc mulţumiri domnilor Prof. dr. ing. Dan Stoiciu şi Prof. dr. ing. Traian Jurca, ale căror sfaturi şi încurajări m-au ajutat în perioada desfăşurării programului doctoral.

Le sunt recunoscătoare colegilor din Departamentul de Măsurări şi Electronică Optică pentru atmosfera de lucru plăcută şi sprijinul primit pe parcursul elaborării tezei.

Nu în ultimul rând, mulţumesc familiei mele pentru sprijinul, răbdarea şi dragostea cu care m-au înconjurat în timpul desfăşurării programului doctoral.

Timişoara, octombrie 2012 Ing. Liliana Stoica

Stoica, Liliana

Contribuţii la prelucrarea numerică a semnalelor cu funcţii spline

Teze de doctorat ale UPT, Seria 7, Nr. 52, Editura Politehnica, 2012, 141 pagini, 51 figuri, 14 tabele.

ISSN: 1842-7014

ISBN: 978-606-554-558-8

Cuvinte cheie: funcţii B-spline, interpolare generalizată, interpolare spline cubică, prelucrare numerică a semnalelor, prelucrarea semnalelor în timp real

Rezumat,

În ultimele decenii, în domeniul prelucrării imaginilor s-au dezvoltat numeroase metode de interpolare cu funcţii spline, în contextul conceptului de interpolare generalizată. Plecând de la acestea, teza de doctorat prezintă noi algoritmi de interpolare spline cubică utilizaţi pentru prelucrarea semnalelor unidimensionale. Avantajul unora dintre aceştia este că pot fi implementaţi prin filtre numerice cu răspuns finit la impuls simetrice. Unul dintre noii algoritmi este implementat pe un sistem cu procesor numeric de semnal şi poate fi utilizat pentru prelucrarea semnalelor în timp real. Procesarea datelor în timp real este o cerinţă de bază pentru numeroase aplicaţii din domeniul prelucrării numerice a semnalelor.

Cuprins

Lista de figuri

Lista de tabele

1. INTRODUCERE

1.1. Problematica tezei

Sistemele electronice, atât cele analogice, cât mai ales cele numerice, au pătruns în tot mai multe domenii. Tendinţa este ca procese, de la cele mai simple până la unele complexe, să fie realizate, urmărite şi/sau conduse automat. Sistemele preiau informaţii, le convertesc, le prelucrează şi pe baza lor se iau decizii care pot fi transmise sub formă de comenzi. De multe ori, prelucrarea acestor informaţii se face prin rularea pe sisteme numerice a unor programe specifice. Multe aplicaţii din viaţa cotidiană sunt realizate prin intermediul unor procesoare care execută un program prestabilit. Acestea sunt realizate pe baza unor modele matematice şi a unor algoritmi specifici [86, 88]. Multe fenomene fizice şi sisteme pot fi descrise prin modele matematice. În domenii precum electronica, mecanica, fizica sau chimia este foarte importantă alegerea modelului şi a algoritmului matematic de lucru.

În prelucrarea numerică a semnalelor sunt esenţiale alegerea modelului matematic şi stabilirea algoritmilor specifici. Dezvoltarea acestora se bazează pe cunoaşterea metodelor şi a uneltelor matematice ce pot fi utilizate în acele cazuri şi care pot duce la rezultate cât mai apropiate de cele aşteptate. Stabilirea de noi teorii şi direcţii în matematică dau posibilitatea îmbunătăţirii metodelor de lucru, realizării unor modele mai apropiate de cele reale, a unor algoritmi mai rapizi şi a unor soluţii competitive. Se cere viteză şi calitate la costuri minime. Se lucrează atât la partea hard a sistemelor, dezvoltând tehnologii rapide, cât şi la partea de programare [22, 70]. Astfel, se caută elaborarea unor algoritmi ce asigură rezultate bune prin efectuarea unui număr cât mai mic de operaţii.

Procedeele de reconstrucţie şi interpolare au un rol important în prelucrarea semnalelor. De cele mai multe ori, semnalele sunt preluate prin intermediul unor senzori şi traductoare şi convertite în şiruri de date. Acestea trebuie să conţină informaţia utilă din semnalul iniţial. Semnalul trebuie să poată fi reconstruit pe baza acestor date şi prin interpolare să furnizeze valori ale semnalului şi în puncte intermediare celor în care s-a făcut prelevarea. Acest lucru este valabil atât în cazul semnalelor unidimensionale (ce prezintă variaţie în domeniul timp sau în domeniul frecvenţe), cât şi în cazul celor multidimensionale (imagini reprezentate pe axele XY, sau pentru care se reţin informaţii de culoare).

Abordarea problemei eşantionării prin prisma unor noi clase de funcţii bază (wavelet şi spline) a dus la dezvoltarea conceptului de interpolare generalizată [9, 24, 58, 102, 103]. Acesta a permis elaborarea a numeroase metode de interpolare implementate prin tehnici de filtrare numerică. Acestea sunt folosite, mai ales, în aplicaţii de prelucrare numerică a semnalelor precum redimensionarea imaginilor (mărire sau micşorare), aplicarea unor transformări geometrice, compresia, detecţia conturului [47, 71, 96, 100, 107-116]. În general, imaginile sunt reprezentate printr-un volum mare de date. Pentru prelucrarea lor sunt necesari algoritmi de complexitate redusă, cât mai uşor de implementat şi care necesită timpi de execuţie cât mai mici. În ultimii 20 de ani, foarte multe aplicaţii din acest domeniu utilizează

pentru interpolare funcţiile spline sub diverse forme. Rezolvarea matriceală a problemei de interpolare cu funcţii spline este complexă, necesită timp de calcul mare şi utilizarea multor resurse [60, 112]. Abordarea acestei probleme prin tehnici de filtrare numerică duce la scăderea semnificativă a complexităţii algoritmilor, la prelucrare unui volum mare de date în timp scurt şi la folosirea unor resurse minime.

În literatura de specialitate se fac referiri la posibilitatea de a utiliza aceşti algoritmi în prelucrarea semnalelor unidimensionale, însă nu sunt prezentate şi exemple [100, 111]. Un mare dezavantaj al acestor algoritmi este că nu pot fi utilizaţi pentru prelucrări de semnale în timp real [123].

În acest context, autoarea tezei a realizat noi algoritmi de interpolare care combină tehnici cunoscute într-o nouă manieră [50-55, 89-91]. Algoritmii sunt implementaţi prin filtre IIR sau FIR simple şi pot fi utilizaţi şi pentru aplicaţii în timp real [56].

1.2. Structura tezei

Lucrarea cuprinde 7 capitole a căror structură este prezentată în continuare.

Capitolul 1 prezintă problematica şi structura tezei. Sunt trecute în revistă principalele subiecte care sunt tratate în capitolele următoare.

În capitolul 2 sunt prezentate câteva noţiuni de bază din prelucrarea semnalelor privind reprezentarea, eşantionarea, interpolarea şi filtrarea semnalelor. Sunt redate pe scurt câteva principii care stau la baza teoriei eşantionării în abordarea tradiţională şi a teoriei eşantionării generalizate. Strâns legat de acestea este procesul de interpolare a datelor. Sunt prezentate câteva metode de interpolare dintre cele mai cunoscute, al căror dezavantaj este faptul că pot produce oscilaţii importante între nodurile funcţiei. De aici derivă o nouă abordare a problemei interpolării, prin prisma conceptului de interpolare generalizată, utilizând funcţii spline sau wavelet.

Capitolul 3 este o trecere în revistă a modului în care funcţiile spline au fost utilizate pentru interpolare. Se definesc, pe baza literaturii consultate, funcţiile spline polinomiale şi B-spline şi se evidenţiază proprietăţile care fac ca acestea să fie des utilizate pentru interpolări de bună calitate. Este prezentată metoda matriceală de rezolvare a problemei de interpolare spline cubică. Se trece apoi la tehnicile moderne de interpolare ce folosesc diverse tipuri de funcţii spline, aplicate în special în prelucrarea imaginilor. Este descris un algoritm general de interpolare care utilizează funcţii B-spline şi tehnici de filtrare numerică preluat din literatura de specialitate. Pornind de la acest algoritm, mai mulţi cercetători au dezvoltat o serie de alte metode de interpolare bazate pe alte tipuri de funcţii spline, sau funcţii spline modificate. În aproape toate aplicaţiile studiate se propun abordări diferite ale pasului doi din procesul de interpolare generalizată. Pentru primul pas, cel de prefiltrare, se păstrează acelaşi tip de filtru IIR folosit în algoritmul iniţial. Tot aici este prezentat în detaliu *algoritmul rapid de interpolare spline cubică* (*algoritmul Unser*). Autoarea tezei a implementat acest algoritm prin programe în *Matlab* şi l-a testat pe câteva tipuri de semnale electrice bine cunoscute. Tot în Capitolul 3 sunt prezentate câteva dintre rezultatele acestor teste şi concluziile ce derivă din acestea, concluzii referitoare în special la filtrul IIR folosit pentru determinarea coeficienţilor B-spline.

În capitolul 4 sunt prezentaţi noi algoritmi de interpolare spline cubică dezvoltaţi de autoare pe baza algoritmului rapid de interpolare spline cubică.

Metodele de realizare a pasului de prefiltrare sunt originale, păstrându-se din algoritmul iniţial partea de filtrare B-spline inversă. Sunt prezentaţi 6 noi algoritmi pentru care coeficienţii B-spline se determină prin metode diferite, metode care derivă din modul de determinare a valorilor derivatelor funcţiei de aproximare în noduri. Sunt analizate câteva rezultate semnificative ale testării acestor algoritmi pe aceleaşi semnale folosite şi în capitolul anterior. *Algoritmii 2D* şi *2P* dezvoltaţi, au marele avantaj că sunt implementaţi prin filtre FIR simetrice.

Capitolul 5 aduce în discuţie câteva metode de îmbunătăţire a algoritmilor dezvoltaţi de autoare, prezentând şi rezultate pentru acestea. Tot în acest capitol este tratată şi problema apariţiei erorilor la capete (erori care la capetele şirului au valori mai mari decât pentru restul eşantioanelor). Aceste erori apar şi în cazul algoritmului preluat din literatură şi se propune o rezolvare originală a acestei probleme. Mai sunt prezentate şi 3 noi metode alternative de implementare a algoritmilor pentru factor de interpolare mai mare ca 2.

În Capitolul 6, autoarea propune implementarea *algoritmului 2D* original pe un sistem cu procesor numeric de semnal. Mai întâi sunt prezentate, pe scurt, caracteristici de bază ale procesoarelor numerice de semnal şi probleme generale legate de lucrul cu acestea. Este descris sistemul utilizat pentru implementarea şi rularea algoritmului. Sunt redate câteva rezultate practice prin intermediul unor imagini importate de la osciloscopul numeric utilizat pentru vizualizarea semnalelor de la intrarea şi respectiv, de la ieşirea sistemului de prelucrare.

Capitolul 7 prezintă cele mai importante concluzii şi contribuţii originale ale autoarei în cadrul tezei de doctorat. Sunt precizate şi posibile dezvoltări ulterioare.

În Anexe sunt redate secvenţe din programele realizate de autoare în *Matlab* şi în *C* (în mediul de dezvoltare *Code Composer Studio*) şi rezultate reprezentative obţinute în urma rulării acestora.

2. EŞANTIONAREA ŞI INTERPOLAREA SEMNALELOR

2.1. Introducere

Semnalele regăsite în aplicaţii sunt în general analogice, deoarece acestea provin de la senzori şi traductoare care transformă variaţia unui fenomen fizic în semnal electric. Se poate face prelucrarea analogică a semnalelor prin aducerea acestora la intrarea unor circuite analogice care realizează operaţii specifice. Prelucrarea numerică a semnalelor analogice se poate face dacă acestea sunt mai întâi convertite în secvenţe de date prin eşantionare şi cuantizare, apoi aduse la intrarea unui sistem numeric de prelucrare (SNP).

Eşantionarea este operaţia de preluare a unor valori ale semnalului la anumite intervale de timp egale (eşantionare uniformă) sau inegale (eşantionare neuniformă). Semnalul în timp contiuu va fi astfel reprezentat în timp discret. Eşantionul este apoi convertit într-un număr prin intermediul unui convertor analog-numeric. Conversia presupune 2 componente: cuantizarea şi codarea valorii cuantizate, convertorul furnizând câte o reprezentare numerică pentru fiecare eşantion [39]. După prelucrarea numerică, semnalul digital poate fi din nou convertit în semnal analogic prin conversie numeric-analogică. Întreg procesul este reprezentat schematic în figura 2.1.

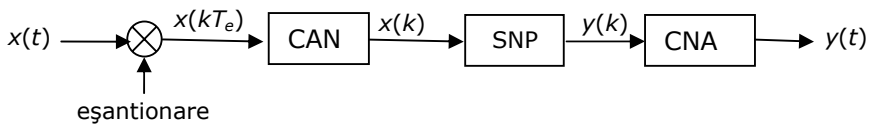


Fig.2.1. Prelucrarea numerică a semnalelor analogice.

În ultimele decenii prelucrarea numerică a semnalelor este folosită în tot mai multe domenii. Aplicaţii complexe care nu pot fi implementate prin intermediul circuitelor analogice sau care necesită costuri mari de implementare sunt realizate prin prelucrare digitală pe sisteme numerice de prelucrare (procesoare numerice de semnal, microprocesoare, calculatoare). Avantajele utilizării sistemelor numerice de prelucrare derivă din caracteristicile acestora. Se pot enumera câteva dintre aceste caracteristici: flexibilitatea în programare, repetabilitatea, stabilitatea semnalelor furnizate, sensibilitatea redusă faţă de perturbaţii [86].

Uzual, semnalele analogice pot fi reprezentate printr-o funcţie $x(t)$ (semnalele deterministe). Semnalele pot fi periodice sau neperiodice, pentru perioadă folosind notaţia T. Semnalele în timp discret vor fi notate $x[k]$ sau $x(k)$, cu k=0, 1,..., N-1, unde N reprezintă numărul de eşantioane al şirului dat.

În domeniul prelucrării semnalelor se utilizează foarte des reconstrucţia şi interpolarea semnalelor. Procesul de interpolare al unui semnal presupune

determinarea funcţiei $x(t)$ pe baza eşantioanelor $x(k)$ prelevate în domeniul timp. Reconstrucţia semnalelor se poate face şi în cazul în care eşantioanele corespunzătoare semnalului $x(t)$ sunt cunoscute în domeniul frecvenţe [69]. Interpolarea se poate implementa prin:

- metode analogice, folosind circuite electronice specifice;
- metode numerice, prin implementarea unor algoritmi de prelucrare numerică pe sisteme numerice de prelucrare.

O serie de operaţii care se efectuează asupra semnalelor numerice se pot implementa prin filtre numerice. În continuare se vor prezenta câteva concepte legate de procesele de eşantionare, interpolare şi filtrare.

2.2. Eşantionarea semnalelor

Fie un semnal analogic $x(t)$, k un număr întreg şi T_e perioada de eşantionare (pasul de eşantionare). Un semnal analogic $x(t)$ este eşantionat periodic la intervale egale cu T_e, rezultând şirul:

$$x(k)=x(kT_e), \quad -\infty>k>\infty \tag{2.1}$$

Perioadei de eşantionare îi corespunde o frecvenţă (rată) de eşantionare:

$$f_e = 1/T_e \tag{2.2}$$

Eşantionarea şi reconstrucţia semnalelor sunt operaţii fundamentale în procesarea numerică a semnalelor. Problema esenţială care se pune este aceea că un semnal eşantionat trebuie să păstreze toate caracteristicile semnalului analogic din care provine pentru ca acesta din urmă să poată fi reconstituit din eşantioanele sale. La baza dezvoltării principiilor eşantionării şi reconstrucţiei semnalelor stă teorema lui Shannon, publicată în 1940 [101]. Mai este cunoscută şi ca teorema WKS: Whittaker-Kotelnikov-Shannon. Aceasta descrie o metodă generală de transformare a unui semnal analogic într-o secvenţă numerică [101]. Prin teorema lui Shannon se spune că pentru a putea reconstrui un semnal din eşantioanele sale, acesta trebuie eşantionat cu o frecvenţă de cel puţin 2 ori mai mare decât cea mai mare frecvenţă din spectrul semnalului. În literatura de specialitate mai este cunoscută şi ca frecvenţă de eşantionare Nyquist [86, 101, 106]. Se presupune că semnalele sunt de bandă limitată, iar pentru reconstrucţie avem nevoie de o infinitate de termeni. Dacă spectrul semnalului conţine şi frecvenţe superioare frecvenţei Nyquist, atunci apare fenomenul de aliere. Pentru a evita apariţia acestui fenomen, se utilizează un filtru trece-jos ideal pentru filtrarea semnalului înainte de eşantionare. Acesta este denumit filtru anti-aliere.

Pentru reconstrucţia unui semnal analogic trebuie de fapt să determinăm funcţia care descrie forma acestui semnal pe baza eşantioanelor prelevate în domeniul timp sau a celor din domeniul frecvenţe obţinute în urma unui proces de prelucrare. Semnalul $x(t)$ de bandă limitată poate fi reconstruit din eşantioanele sale $x(kT_e)$ după formula:

$$x(t) = \sum_{k \in Z} x(kT_e)\, sin\, c(t/T_e - k) \tag{2.3}$$

unde *sinc*(*t*) este funcţia *sinus cardinal*:

$$sin\,c(t) = \frac{sin(\pi t)}{(\pi t)} \tag{2.4}$$

Pentru multă vreme, teorema eşantionării a lui Shannon a reprezentat singurul mod de abordare a problemei eşantionării. În 1977 Papoulis a demonstrat că, pentru un semnal de bandă limitată, este posibilă reconstituirea semnalului din eşantioanele obţinute la ieşirea unui ansamblu de q sisteme liniare invariante în timp dacă eşantionarea s-a făcut cu o frecvenţă de $(1/q)$ din frecvenţa Nyquist [42, 117]. Astfel se realizează o eşantionare multicanal, neideală, oferind o abordare mai realistă a problemei decât în cazul eşantionării ideale propusă de teorema lui Shannon. Această generalizare a pornit de la faptul că există multiple posibilităţi de a obţine date dintr-un semnal pentru o caracterizare completă, eşantionarea uniformă cu frecvenţa Nyquist fiind doar o posibilitate.

În principiu, teorema generalizată a lui Papoulis propune o *eşantionare multicanal* care poate fi un instrument util în practică, în special în situaţii de achiziţie de semnal cu senzori multipli. Papoulis a formulat această teorie pentru funcţii de bandă limitată, însă nu a realizat un algoritm concret care ar putea fi implementat.

Eşantionarea multicanal se poate extinde şi în cazul în care semnalele de intrare nu sunt de bandă limitată [42, 43, 77, 78, 117, 118]. Semnalul de intrare nu se mai consideră de bandă limitată, ci este un semnal de energie finită: $x(t) \in L_2$. Se renunţă la idea de reconstrucţie exactă a semnalului de intrare, făcând o aproximare „consistentă" a acestuia.

Scopul eşantionării este acela de a reprezenta o funcţie $x(t)$ de variabilă continuă t printr-un şir de numere, reprezentare care este de cele mai multe ori preferabilă pentru prelucrarea şi transmiterea de date. Reprezentarea nu trebuie să fie ambiguă, de aceea problema se limitează la o anumită clasă dată de semnale. În teoria clasică se consideră spaţiul semnalelor de bandă limitată pentru care funcţia bază este funcţia sinus cardinal.

Teorema eşantionării a lui Shannon descrie legătura dintre o funcţie de bandă limitată şi eşantioanele sale echidistante prelevate la o frecvenţă superioară frecvenţei Nyquist. Această abordare prezintă mai multe probleme din punct de vedere practic:

- presupune utilizarea unor filtre ideale, imposibil de realizat fizic;
- semnalele cu care se lucrează în practică sunt de durată finită, ceea ce este în contradicţie cu ipoteza de semnal de bandă limitată;
- limitarea benzii de frecvenţă tinde să genereze oscilaţii Gibbs;
- funcţia bază – sinus cardinal – are o rată de cădere foarte mică.

Primele 2 probleme pot fi oarecum rezolvate prin aproximări, însă ultimele două se pot rezolva doar prin schimbarea funcţiei bază [31, 101]. O abordare recentă este aceea de a studia eşantionarea prin metode oferite de teoria multirezoluţiei şi a transformatei Wavelet. Baza acestei abordări o constituie faptul că diverse subspaţii wavlet au aceeaşi structură invariantă în timp ca şi clasa funcţiilor de bandă limitată Shannon. Teoria funcţiilor wavelet a fost cea care a adus un nou mod de gândire în procesarea semnalelor în termeni de funcţii de bandă nelimitată [33, 104]. Astfel, a fost deschis drumul spre folosirea şi a altor funcţii (de exemplu, funcţiile spline) ca funcţii bază în teoria prelucrării semnalelor. Se oferă o abordare mai generală a problemei eşantionării, iar teoria tradiţională poate fi

considerată un caz particular al acesteia (deoarece atunci când gradul funcţiei B-spline tinde la infinit, aceasta tinde la *sinc*(*t*)).

Orice semnal în timp continuu poate fi descris prin următorul model matematic [120]:

$$x(t) = \sum_{k=-\infty}^{\infty} c(k)\varphi(t-k) \tag{2.5}$$

În abordarea teoremei lui Shannon, coeficienţii $c(k)$ sunt chiar eşantioanele semnalului $x(k)$, iar funcţia $\varphi(t)$ este funcţia *sinus cardinal*. În eşantionarea generalizată, pentru $\varphi(t)$ se folosesc alte clase de funcţii (de exemplu: wavelet sau B-spline), iar coeficienţii sunt o reprezentare a semnalului în timp discret.

Eşantionarea wavelet şi spline reprezintă o alternativă tot mai des utilizată la abordarea clasică. Aceste tehnici pot fi folosite cu succes pentru proiectarea şi folosirea dispozitivelor de achiziţie ne-ideale şi a măsurărilor multicanal [117].

Un tip de eşantionare multicanal ce poate fi utilă, de exemplu, în cazul în care avem un şir de măsurări asupra poziţiei şi vitezei unei ţinte sau a unui obiect în mişcare, este *eşantionarea derivativă* [119]. Dacă avem la dispoziţie semnalul $x(t)$ şi derivata sa de ordinul întâi $x'(t)$, atunci se pot eşantiona fiecare cu o frecvenţă de 2 ori mai mică decât frecvenţa Nyquist. Numărul total de eşantioane pe unitatea de timp este acelaşi ca în cazul eşantionării clasice [119, 120]. Eşantioanele obţinute pot fi scrise astfel:

$$x(2n) = \sum_{k} c(k)\varphi(2n-k) \tag{2.6}$$

$$x'(2n) = \sum_{k} c'(k)\varphi(2n-k) \tag{2.7}$$

Se consideră că acestea sunt obţinute la ieşirea unui sistem de filtre cu 2 canale, filtrele având funcţiile de transfer:

$$H_0(z) = \sum_{n} \varphi(n) z^{-n} \tag{2.8}$$

$$H_1(z) = \sum_{n} \varphi(n) z^{-n} \tag{2.9}$$

Impunând anumite condiţii pentru aceste filtre, din eşantioanele $x(2n)$ şi $x'(2n)$ putem reconstrui perfect $c(k)$, deci în final putem obţine semnalul $x(t)$.

În cazul în care $x(t)$ este o funcţie spline de gradul n, în loc să aplicăm eşantionarea uniformă conform teoriei lui Shannon, putem să eşantionăm $x(t)$ şi cele n-1 derivate ale sale, fiecare cu o frecvenţă de n ori mai mică. Numărul total de eşantioane pe perioadă este acelaşi. În acest caz, $\varphi(t)$ poate fi funcţia B-spline cubică sau funcţia B-spline de ordinul 2. Este demonstrat că $x(t)$ poate fi reconstruit din aceste eşantioane utilizând filtre FIR digitale [119, 120].

În ultimii ani, teoria eşantionării generalizate a dus la dezvoltarea a numeroase metode de eşantionare uniformă sau neuniformă ce se pot aplica pentru semnale nelimitate în bandă şi pentru sisteme neideale [21, 38, 48, 72, 95]. Această teorie este în strânsă legătură cu conceptul de interpolare generalizată.

2.3. Interpolarea semnalelor

Problema interpolării se pune în situaţii care necesită determinarea unor informaţii ce nu se regăsesc în mod explicit într-un şir de date. Fiind date un şir de valori ale unei funcţii necunoscute, prin interpolare se construieşte o funcţie care are aceleaşi valori cu funcţia originală în punctele date şi permite calcularea valorilor şi în alte puncte din domeniul de definiţie.

Diverse metode de interpolare s-au folosit încă din antichitate pentru calcularea timpului şi întocmirea calendarului pe baza poziţiei anumitor stele. Teorii şi metode de interpolare au fost dezvoltate în diferite colţuri ale lumii şi în diferite perioade temporale. Lipsa posibilităţilor de comunicare şi documentare la distanţe lungi a făcut ca în zone geografice diferite, în aceeaşi perioadă de timp sau în perioade diferite să se dezvolte metode similare [58].

Din punct de vedere matematic, problema interpolării se pune astfel: fie o funcţie reală $f(x)$ definită pe un interval $[x_0, x_N]$ pentru care se cunosc valorile funcţiei în punctele x_k, notate $y_k=f(x_k)$. Să se aproximeze această funcţie printr-o altă funcţie $g(x)$, relativ simplă, astfel încât, în orice punct, valoarea celei de-a doua funcţii să fie „suficient de aproape" de valoarea lui f. Punctele x_k în care se dau valorile funcţiei se numesc noduri. Altfel spus, se caută o funcţie $g(x)$ care în noduri ia aceleaşi valori ca funcţia $f(x)$, sau valori apropiate de acestea:

$$g(x_k)= f(x_k),\ \ k=0, 1, ..., N \tag{2.10}$$

iar în celelalte puncte de pe interval, valorile lui $g(x)$ şi $f(x)$ să fie apropiate [60].

Se pot distinge două cazuri în care este necesară aproximarea unei funcţii:

- funcţia are o expresie complicată şi este dificil de evaluat;
- se cunosc valorile funcţiei în câteva puncte şi este necesar să se determine valorile funcţiei în alte puncte intercalate [45].

Una dintre cele mai simple metode de interpolare este *interpolarea liniară*. Metoda de interpolare liniară presupune aproximarea unei funcţii date între 2 puncte prin segmentul de dreaptă ce uneşte cele 2 puncte. Polinomul de aproximare de gradul întâi este de forma:

$$P(x)=\frac{f(x_{k+1})-f(x_k)}{x_{k+1}-x_k}x+\left[f(x_k)-\frac{f(x_{k+1})-f(x_k)}{x_{k+1}-x_k}x_k\right] \tag{2.11}$$

Interpolarea liniară a fost introdusă prima dată în calculul funcţiilor trigonometrice şi al logaritmilor [20, 45]. De obicei, acest tip de interpolare implică erori foarte mari, care depind de interval şi de forma funcţiei pe acest interval. Se poate obţine o anumită acurateţe dacă se utilizează un număr suficient de puncte de interpolare [20]. Din această cauză, interpolarea liniară este rar utilizată, în schimb folosindu-se *interpolarea polinomială* (polinom de interpolare de grad mai mare).

Prin interpolarea polinomială se caută un polinom de aproximare de gradul n care în nodurile de interpolare să coincidă cu valorile funcţiei. S-au dezvoltat multe

metode de a determina acest polinom, printre cele mai cunoscute fiind polinomul lui Newton şi polinomul Lagrange.

Polinomul de interpolare Newton se poate construi pe baza diferenţelor finite ale funcţiei şi se poate folosi doar în cazul în care nodurile de interpolare sunt echidistante. Acesta se determină astfel:

$$N_n(x) = y_0 + \frac{\Delta y_0}{1!\,h}(x - x_0) + \frac{\Delta^2 y_0}{2!\,h^2}(x - x_0)(x - x_1) + \ldots + \\ + \frac{\Delta^n y_0}{n!\,h^n}(x - x_0)(x - x_1)\ldots(x - x_{n-1}) \qquad (2.12)$$

Diferenţele finite sunt utilizate doar în cazul funcţiilor definite în puncte echidistante. Polinoamele de interpolare pot fi generalizate pentru funcţii reprezentate în puncte inegal distanţate, utilizând diferenţe divizate pentru determinarea acestora [45]. Prin metoda Lagrange se propune găsirea unui polinom de aproximare pentru funcţii ale căror valori sunt date în puncte inegal distanţate. *Polinomul Lagrange* se determină astfel [45, 75]:

$$L_n(x) = \sum_{k=0}^{n} f(x_k) \frac{(x - x_0)(x - x_1)\ldots(x - x_{k-1})(x - x_{k+1})\ldots(x - x_n)}{(x_k - x_0)(x_k - x_1)\ldots(x_k - x_{k-1})(x_k - x_{k+1})\ldots(x_k - x_n)} \qquad (2.13)$$

Deşi valoarea polinomul coincide cu cea a funcţiei în noduri, între două noduri poate să se abată foarte mult de la aceasta. Acesta este marele dezavantaj al folosirii polinoamelor de interpolare. În plus, ori de câte ori se doreşte să se înlocuiască polinomul cu unul de grad mai mare, toate calculele de determinare a acestuia trebuie refăcute [45].

Într-o serie de aplicaţii, pe lângă faptul că funcţia de aproximare trebuie să furnizeze aceleaşi valori ca şi funcţia de aproximat în punctele de interpolare, se cere ca şi derivatele acestora (de un anumit ordin) să coincidă în anumite puncte. Acest caz, în care sunt date valorile funcţiei şi ale unui număr de derivate consecutive în noduri, se numeşte *interpolare Hermite* (*hermitiană*) [60].

În cazul *interpolării Hermite* se pune problema găsirii unui polinom unic de grad m sau mai mic, care, pentru funcţia de interpolare $H(x_k)$, trebuie să satisfacă condiţiile:

$$H(x_k) = f(x_k), \text{ pentru } k=0, 1, 2, \ldots, N \qquad (2.14)$$

$$H^{(i)}(x_k) = f^{(i)}(x_k), \; i=1, 2, \ldots, m \qquad (2.15)$$

Dacă se scriu toate relaţiile care rezultă din condiţiile de mai sus, se obţine un sistem de $m+1$ ecuaţii liniare cu $m+1$ necunoscute, sistem ce poate fi rezolvat prin diverse metode [20].

Metodele de interpolare polinomială au avantajul că oferă o formă analitică care poate fi uşor manipulată matematic. Marele dezavantaj al acestora este că aproximarea poate produce oscilaţii considerabile între noduri [75]. Pentru reducerea oscilaţiilor se pot alege mai multe noduri, ceea ce înseamnă construirea unui polinom de grad mai mare. Aceasta nu conduce neapărat la rezultate mai bune

[60]. S-a demonstrat că, dacă gradul polinomului de interpolare creşte indefinit ($n\to\infty$), atunci şi eroarea de interpolare tinde la infinit ($\varepsilon\to\infty$) [60].

S-a căutat o altă clasă de funcţii de interpolare care să înlăture neajunsurile polinoamelor de interpolare cunoscute. Astfel, se face trecerea la o clasă extinsă de funcţii polinomiale: funcţii polinomiale pe porţiuni (sau segmentar polinomiale) – polinoame posibil diferite pe subdomenii diferite ale domeniului pe care se face interpolarea. Segmentele de polinoame se racordează în noduri împreună cu un anumit număr de derivate ale acestora. Acestea sunt denumite funcţii spline [60].

Funcţiile spline cubice sunt cele mai simple şi mai des utilizate în ultimii ani în aplicaţii de interpolare. Aceste funcţii vor fi descrise pe larg în Capitolul 3.

În teoria prelucrării semnalelor, procesele de reconstrucţie şi interpolare sunt direct legate de modul în care s-a făcut eşantionarea semnalelor. În cazul ideal, formula de interpolare care permite reconstituirea semnalului continuu $x(t)$ din eşantioanele sale este dată de relaţia (2.5), unde $\varphi(t)$ este funcţia de interpolare. În practică se lucrează cu un număr finit de eşantioane şi nu se poate implementa această relaţie, care reprezintă un proces de filtrare ideală. De aceea, filtrul ideal va fi înlocuit cu un filtru numeric specializat (filtru de interpolare). Acesta trebuie să îndeplinească următoarele condiţii [69]:

- să prezinte o caracteristică fază – frecvenţă nulă sau cel mult liniară;
- eşantioanele iniţiale trebuie să se reproducă fără erori în şirul interpolat;
- funcţia de transfer a filtrului trebuie să fie o funcţie pară;
- să asigure conservarea componentei continue a semnalului reconstituit.

Se poate spune că filtrele cu răspuns finit la impuls, care sunt filtre nerecursive, sunt potrivite pentru a fi utilizate ca filtre de interpolare.

Conceptul de eşantionare generalizată îl are ca şi corespondent pe cel de interpolare generalizată. În cazul interpolării generalizate se renunţă la idea de interpolare exactă a unui semnal, preferând o bună aproximare a acestuia [12]. Funcţia de interpolare $\varphi(k)$ este înlocuită cu alte funcţii de sinteză care asigură o aproximare consistentă a semnalului iniţial [9, 30]. Idea utilizării unui filtru anti-aliere din teoria clasică a eşantionării a sugerat soluţii asemănătoare şi în cazul interpolării generalizate. Astfel, din eşantioanele cunoscute ale semnalului $x(k)$, printr-un proces de prefiltrare, se determină coeficienţii $c(k)$, cu ajutorul cărora se pot calcula eşantioanele semnalului interpolat [9]. Procesul de interpolare generalizată poate fi descris prin schema din figura 2.2.

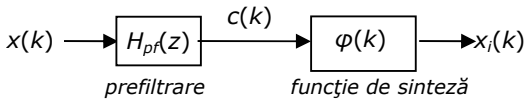


Fig.2.2. Interpolarea generalizată.

Funcţia de transfer a sistemului de prefiltrare se determină astfel [9]:

$$H_{pf}(z)=\frac{1}{\sum_{k\in Z}\varphi(k)z^{-k}} \tag{2.16}$$

În domeniul prelucrării imaginilor, în ultimii ani, s-au dezvoltat şi se utilizează algoritmi de interpolare care se bazează pe conceptul de interpolare generalizată [58, 59, 124]. Pentru $\varphi(k)$ se folosesc funcţiile B-spline de diferite grade, sau variante modificate ale acestora. Câteva dintre aceste metode vor fi prezentate pe scurt în Capitolul 3.

2.4. Filtrele numerice

Filtrele numerice sunt sisteme liniare invariante utilizate pentru a modifica distribuţia în frecvenţă a componentelor unui semnal după specificaţii date, folosind operaţii aritmetice cu o acuratețe limitată. Dacă la intrarea unui astfel de sistem se aduce un semnal discret $x(k)$, operaţia care se efectuează asupra lui se numeşte filtrare numerică şi la ieşirea sistemului se obţine $y(k)$.

Procesul de filtrare este descris prin relaţia (2.17), unde $h(i)$ este răspunsul sistemului la impulsul unitate.

$$y(k) = \sum_{i=0}^{N} h(i) \cdot x(k-i) \tag{2.17}$$

Pentru relaţia (2.17) se aplică transformata Z şi se obţine:

$$Y(z) = H(z)X(z) \tag{2.18}$$

Orice filtru numeric este caracterizat prin funcţia de transfer notată $H(z)$. Aceasta este dată ca raport de polinoame, termenii a_i şi b_i fiind coeficienţii filtrului:

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_N z^{-N}}{1 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_M z^{-M}} = \frac{A(z)}{B(z)} \tag{2.19}$$

Prin urmare, semnalul de la ieşirea filtrului se poate determina astfel:

$$y(k) = \sum_{i=o}^{N} a_i x(k-i) - \sum_{j=0}^{M} b_j y(k-j) \tag{2.20}$$

După tipul de răspuns la impulsul unitate, filtrele se pot clasifica în:

- filtre cu răspuns finit (FIR) la impulsul unitate;
- filtre cu răspuns infinit (IIR) la impulsul unitate.

Pentru filtrele cu răspuns finit la impuls (FIR) coeficienţii b_i sunt zero, funcţia de transfer a filtrului fiind dată prin polinomul $A(z)$. Acestea sunt filtre nerecursive. Un astfel de filtru este întotdeauna stabil, deoarece are răspuns finit la impulsul unitate. Dacă filtrul este simetric (şirul coeficienţilor a_i este simetric), atunci caracteristica de fază este o funcţie liniară cu frecvenţa [39, 86].

Filtrele cu răspuns infinit la impuls (IIR) sunt caracterizate printr-o funcţie de transfer de forma celei din relaţia (2.19), unde polinomul $B(z)$ este diferit de

unu. Funcţia de transfer are *N* zerouri şi *M* poli (p_i). Pentru ca filtrul să fie stabil, trebuie ca polii funcţiei de transfer să se situeze în planul *z* în interiorul cercului unitate, iar modulul acestora să fie mai mic decât unitatea. Ca filtrul să fie cauzal trebuie ca numărul polilor să fie mai mare decât numărul zerourilor [39, 64, 86]. Putem aprecia stabilitatea unui filtru IIR astfel:

- dacă $|p_i|<1$, atunci $h(i)\to 0$, rezultând un sistem stabil;
- dacă $|p_i|>1$, atunci $h(i)\to\infty$, rezultând un sistem instabil;
- dacă $|p_i|=1$, sistemul este caracterizat de o stabilitate marginală şi produce un răspuns oscilatoriu. Mai mult, polii de ordin multiplu de pe cercul unitar determină un sistem instabil.

Filtrele IIR sunt filtre recursive, pentru care valoarea semnalului de la ieşire la un moment dat depinde atît de valorile semnalului de la intrare cât şi de valorile anterioare ale semnalului de la ieşire.

Pentru filtre de ordin mai mare, dacă funcţia de transfer $H(z)$ dată de (2.19) poate fi împărţită conform relaţiei (2.21), atunci filtrul poate fi obţinut prin legarea în cascadă a *r* filtre de ordin mai mic, care au funcţiile de transfer $H_1(z)$, ..., $H_r(z)$.

$$H(z)=C\cdot H_1(z)\cdot H_2(z)\cdot\ldots\cdot H_r(z) \tag{2.21}$$

În figura 2.3 este prezentată structura în cascadă a unui filtru.

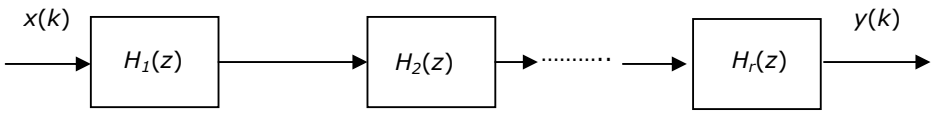


Fig. 2.3. Structura în cascadă.

Implementarea filtrelor numerice presupune realizarea unor operaţii simple de adunare şi înmulţire. De aceea, ele pot fi implementate cu uşurinţă pe sisteme numerice de prelucrare. Procesoarele numerice de semnal sunt optimizate să poată efectua în paralel cel puţin o operaţie de înmulţire şi una de adunare, realizându-se astfel uşor şi rapid procesele de filtrare numerică.

3. METODE DE INTERPOLARE CU FUNCŢII SPLINE

3.1. Introducere

Funcţiile spline reprezintă o clasă specială de funcţii dezvoltată din necesitatea de a elimina unele neajunsuri ale funcţiilor polinomiale în teoria aproximării şi interpolării funcţiilor. Funcţia este formată din polinoame pe subintervale adiacente care se racordează în noduri împreună cu un anumit număr de derivate ale sale.

Denumirea acestui tip de funcţii provine din mecanică. Dispozitivul mecanic „spline" este format dintr-o bară subţire deformabilă cu greutăţi care pot fi aranjate astfel încât bara să treacă prin anumite puncte date şi este utilizat pentru a desena o curbă netedă [60]. În figura 3.1 este prezentat un astfel de instrument [130].

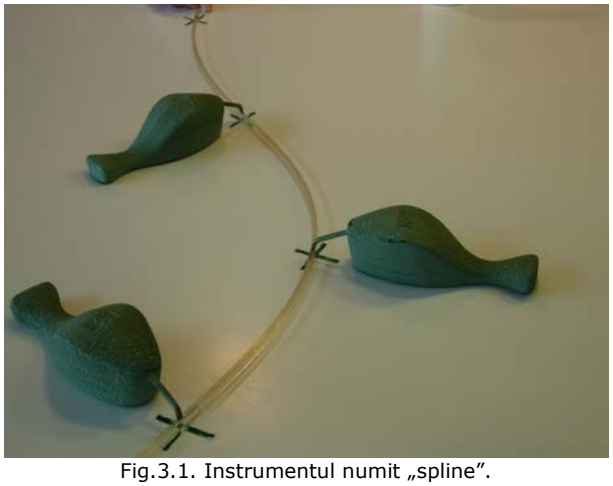


Fig.3.1. Instrumentul numit „spline".

Chiar din antichitate, liniile poligonale au fost utilizate în problemele de geometrie, apoi pentru aproximarea soluţiilor ecuaţiilor diferenţiale. Isaac Jacob Schoenberg(1) a utilizat pentru prima dată în 1946 termenul de „funcţie spline" pentru a descrie o funcţie segmentar polinomială [60, 100]. El a avut un rol important în structurarea şi dezvoltarea teoriei funcţiilor spline, fiind cunoscut ca „tatăl" acestor funcţii [130].

Carl DeBoor este considerat ca fiind cel care a continuat munca lui Schoenberg în domeniul funcţiilor spline [130].

Dezvoltarea teoriei funcţiilor spline s-ar putea împărţi în 2 perioade [60, 100]. Prima este până în 1964, când un număr însemnat de matematicieni au

(1) I. J. Schoenberg s-a născut în România, la Galaţi, în 1903. A studiat şi apoi a fost asistent la *Universitatea din Iaşi*, unde a susţinut şi teza de doctorat. După 1930, activitatea sa de cercetare se desfăşoară în Statele Unite ale Americii. Lucrările sale sunt de referinţă în dezvoltarea ulterioară a teoriei funcţiilor spline [82].

dezvoltat anumite clase de funcţii care aveau proprietăţi asemănătoare sau identice cu funcţiile spline. Între timp, teoria prelucrării semnalelor a cunoscut o dezvoltare rapidă datorită abordării prin prisma funcţiilor de bandă limitată propusă de Shannon [100]. În 1964, I. J. Schoenberg a descoperit relaţia dintre funcţiile spline de interpolare şi cele mai bune formule de aproximare în sensul lui Sard. Se poate spune că acesta a fost un pas hotărâtor în dezvoltarea teoriei funcţiilor spline [60]. După aceasta, interesul pentru acest domeniu a crescut şi tot mai mulţi au fost cei care au dezvoltat şi utilizat teoriile existente.

Ulterior, familia funcţiilor spline este extinsă, incluzând funcţii spline polinomiale, funcţii spline trigonometrice, funcţii spline exponenţiale, B-spline, L-spline, funcţii spline Cebîşev, funcţii spline cardinale, biortogonale. În literatura de specialitate sunt tratate funcţiile spline de o variabilă, teoria fiind extinsă rapid şi în cazul mai multor variabile [60, 16]. Teoria funcţiilor spline a fost dezvoltată într-o mulţime de direcţii şi mai ales, s-a căutat aplicarea funcţiilor spline în diverse domenii. În matematică şi în ramurile care necesită un suport şi un model matematic bine elaborat, aceste funcţii se utilizează în rezolvarea de probleme noi, sau oferă o nouă abordare a problemelor clasice (interpolare, aproximarea unor funcţionale liniare, rezolvarea numerică a unor clase de ecuaţii diferenţiale şi integrale, ecuaţii cu derivate parţiale) [60, 100].

Odată cu dezvoltarea computerelor, funcţiile spline au avut un impact considerabil în proiectarea asistată de calculator şi grafica computerizată [100]. În prelucrarea semnalelor au fost folosite mai târziu, după ce teoria funcţiilor wavelet a adus un nou mod de gândire prin prisma funcţiilor de bandă nelimitată [100].

Se poate spune că rezultate importante în teoria funcţiilor spline s-au obţinut şi în România. Membrii seminarului de cercetare „Funcţii spline şi aplicaţii" de la *Facultatea de matematică* a *Universităţii din Cluj-Napoca* au publicat lucrări în acest domeniu. O lucrare de bază este monografia *„Funcţii spline şi aplicaţii"* [60] a profesorului Gheorghe Micula. În numeroase rânduri se vor face referiri la această carte şi din ea vor fi prezentate câteva definiţii şi teoreme în subcapitolul 3.2. Lucrarea, revizuită şi completată, scrisă în limba engleză împreună cu Sanda Micula, este publicată în 1999 sub titlul *„Handbook of splines"* [61].

3.2. Funcţii spline polinomiale – definiţii şi proprietăţi

Funcţiile spline polinomiale vor fi definite în continuare.

Definiţie 3.1 [60]: Funcţia $s:\Re\to\Re$ se numeşte *funcţie spline de gradul n*, cu nodurile $x_1< x_2<\ldots<x_N$, dacă satisface următoarele două condiţii:

- $s\big|_{I_i} \in P_n(I_i), I_i = (x_i, x_{i+1}), i = 0,1,\ldots,N, (x_0 = -\infty, x_{N+1} = +\infty)$; (3.1)
- $s \in C^{n-1}(\Re)$. (3.2)

O funcţie spline de gradul *n,* definită pe un interval, este formată din fragmente de polinoame de grad *n* pe subintervale adiacente, care se unesc în noduri. Funcţia este de *n*-1 ori derivabilă, ea şi primele *n*-1 derivate ale sale fiind continue. Datorită proprietăţilor de continuitate şi a faptului că se exprimă prin polinoame, acest tip de funcţie poate fi uşor de folosit în rezolvarea problemelor de interpolare a datelor şi aproximare a unor curbe şi funcţii complicate. Acestea nu sunt necesare în cazuri în care este simplu de determinat funcţia ce se potriveşte

unui set de date, ci pentru a genera o funcţie care să le aproximeze. Se pot realiza uşor algoritmi de calcul ce pot fi implementaţi prin programe pe calculator sau pe procesoare numerice de semnal.

Următoarea teoremă arată faptul că funcţiile spline polinomiale pot fi folosite pentru aproximare.

Teoremă 3.1 [60]: Orice funcţie $f \in C^n[a, b]$, împreună cu derivatele ei până la ordinul n, pot fi aproximate uniform printr-o funcţie spline de gradul n, respectiv prin derivatele până la ordinul n ale funcţiei spline.

În problemele de interpolare, se utilizează frecvent funcţiile spline de grad impar, deoarece acestea au anumite proprietăţi deosebite (de exemplu: estimarea erorii este mai bună) [60]. Se demonstrează şi faptul că este avantajos din punct de vedere practic dacă condiţiile de interpolare se pun chiar în noduri. Acest lucru este indicat în cazul funcţiilor de grad impar. Pentru funcţiile spline de grad par se recomandă ca nodurile funcţiei de interpolare să nu coincidă cu punctele în care sunt cunoscute datele de intrare [16, 19, 44].

Cel mai des sunt utilizate funcţiile spline cubice. Funcţiile spline cubice sunt formate din polinoame de gradul 3.

Definiţie 3.2 [60]: Funcţia $s_\Delta:[a, b] \to \Re$ se numeşte *funcţie spline cubică*, relativ la diviziunea Δ (cu nodurile x_i, i=1, 2, ... , N-1), dacă satisface următoarele două condiţii:

- $s_\Delta \in P_3(I_i), i = 0,1,...,N-1$; (3.3)
- $s_\Delta \in C^2[a,b]$. (3.4)

unde: Δ - reprezintă o diviziune a intervalului $[a, b]$;
x_i - puncte din intervalul $a = x_1 < x_2 < \ldots < x_N = b$;
$I_i = (x_i, x_{i+1})$ – subintervale;
$P_3(I_i)$ – restricţia tuturor polinoamelor de grad≤3 pe intervalul I_i.

Definiţie 3.3 [60]: Fie $Y=(y_1, y_2, ..., y_N)$ un vector dat. O funcţie spline cubică s_Δ se numeşte *funcţie spline cubică de interpolare* pentru vectorul Y, pe diviziunea Δ, dacă:

$$s_\Delta(x_i) = y_i, \quad \text{pentru } i=1, 2, ..., N \tag{3.5}$$

Dacă Y reprezintă valorile unei funcţii date f în noduri, atunci s_Δ este o funcţie spline cubică de interpolare pentru funcţia f şi se notează: s_f sau s_y. Micula demonstrează existenţa şi unicitatea acestei funcţii de interpolare [60].

Funcţiile spline cubice de interpolare au proprietăţi de convergenţă foarte bune. Dintre toate funcţiile $g:[a, b] \to \Re$ care interpolează funcţia dată f, funcţia spline cubică are proprietatea remarcabilă de a minimiza integrala:

$$\int_a^b [g''(x)]^2 dx \tag{3.6}$$

Aceste proprietăţi pot fi îmbunătăţite prin cerinţe mai tari de continuitate, de exemplu pentru o funcţie $f \in C^4[a, b]$ [60]. Aceste rezultate sunt în contrast cu rezultatele negative asupra convergenţei polinoamelor de interpolare, însă, pentru

clase şi mai netede de funcţii, ordinul de convergenţă nu poate creşte şi mai mult [60].

În practică există cazuri în care valorile y_i în nodurile x_i nu sunt cunoscute cu exactitate. De exemplu, acestea sunt rezultate determinate experimental, însoţite de o anumită incertitudine de măsurare. În aceste cazuri, problema interpolării presupune găsirea unei funcţii spline care în noduri să ia valori aproximative, nu valori exacte: $s_y(x_i) \cong y_i$. O astfel de funcţie se numeşte *funcţie spline de ajustare* [60].

Definiţiile funcţiilor spline polinomiale şi metodele de determinare a funcţiilor spline de interpolare prezentate se regăsesc în mai multe lucrări din literatura de specialitate [16, 17, 44, 93].

3.3. Metoda clasică de interpolare cu funcţii spline

Se dă vectorul de date $Y=(y_1, y_2,..., y_N)$. Trebuie să determinăm funcţia spline cubică s cu nodurile în punctele x_i, $i=1,...,N$, unde $a= x_1< x_2<\ldots<x_N = b$ este o diviziune a intervalului $[a,b]$, funcţie care să interpoleze aceste date.

Funcţiile spline cubice pot fi exprimate astfel:

$$s(x) = \begin{cases} s_1(x), & \text{dacă} \quad x_1 \le x < x_2 \\ s_2(x), & \text{dacă} \quad x_2 \le x < x_3 \\ \vdots & \\ s_{N-1}(x), & \text{dacă} \quad x_{N-1} \le x < x_N \end{cases} \tag{3.7}$$

unde $s_i(x)= a_i (x-x_i)^3+ b_i (x-x_i)^2 +c_i (x-x_i)+d_i$, pentru $i=1,2,\ldots,N-1$. (3.8)

Derivatele întâi şi a doua ale funcţiei vor fi:

$$s_i'(x)= 3a_i (x-x_i)^2+2b_i (x-x_i) +c_i \tag{3.9}$$

$$s_i''(x)= 6a_i (x-x_i)+2b_i \tag{3.10}$$

Corespunzător celor N-1 subintervale ale funcţiei, avem N-1 fragmente de polinoame de gradul 3, caracterizate fiecare prin 4 parametri (a_i ,b_i ,c_i ,d_i), în total, 4N-4 parametri ce trebuie determinaţi. Prima condiţie care se pune este ca funcţia să treacă prin toate punctele, deci în noduri să ia valorile datelor din vectorul de intrare:

$$s(x_i)= y_i \text{ , } i=1,...,N \tag{3.11}$$

Din aceste relaţii rezultă N ecuaţii.

Funcţia şi derivatele sale de ordinul 1 şi 2 trebuie să fie continue pe tot intervalul $[a,b]$, de unde mai rezultă 3(N-2) condiţii:

$$s_i(x_i)= s_{i-1}(x_i) \tag{3.12}$$

$$s_i'(x_i)= s_{i-1}'(x_i) \tag{3.13}$$

$$s_i''(x_i) = s_{i-1}''(x_i) \tag{3.14}$$

În total avem 4*N*-6 condiţii ce reprezintă un sistem de 4N-6 ecuaţii cu 4N-4 necunoscute, deci un sistem nedeterminat.

Cele 2 condiţii de care mai avem nevoie pentru determinarea funcţiei spline cubice în mod unic pot fi specificate în mai multe feluri. De exemplu, *funcţia spline cubică naturală* se obţine pentru condiţii la limită naturale [57, 60, 65]:

$$s''(x_1) = s''(x_N) = 0. \tag{3.15}$$

Dacă ultimele 2 condiţii sunt cele ce care urmează, obţinem *funcţia spline parabolică* [57]:

$$s''(x_1) = s''(x_2) \tag{3.16}$$

$$s''(x_N) = s''(x_{N-1}) \tag{3.17}$$

Funcţia *s* se poate prelungi în afara intervalului [*a*,*b*] prin periodicitate cu perioada *b*-*a*. Astfel se obţin *funcţii spline cubice de interpolare periodice* [65]. În acest caz, condiţiile impuse sunt:

$$s''(x_1) = s''(x_N) \tag{3.18}$$

$$s''(x_2) = s''(x_{N+1}) \tag{3.19}$$

Sistemul permite o rezolvare matriceală [65, 87, 126]. Caracterul polinomial al acestor funcţii prezintă, din punct de vedere aplicativ, avantajul de a putea duce până la capăt calculele şi a determina soluţii unice în condiţii date. Micula demonstrează că problema are soluţie unică [60], dar sunt necesare un număr mare de calcule, fapt ce poate fi dezavantajos din punctul de vedere al implementării soluţiei pe un sistem de prelucrare numerică.

Utilizând şi valorile derivatelor pentru rezolvarea problemei, procesul este unul de interpolare Hermite [60].

În [60], sunt descrise două metode de calcul efectiv al unei funcţii spline cubice de interpolare. Mai întâi defineşte mărimile $m_i = s_y'(x_i)$ (pante) şi respectiv $M_i = s_y''(x_i)$ (momente). Pentru polinomul de interpolare, defineşte şi derivatele acestuia în noduri ca fiind egale cu aceste momente. Aceasta face ca procesul de interpolare să fie unul de tip Hermite. Scriind ecuaţiile pentru polinoame şi derivatele acestora, se ajunge la un sistem de *N*-2 ecuaţii cu *N* necunoscute. Pentru ca sistemul să fie univoc rezolvabil se mai adaugă două condiţii la limită suplimentare. Aceste condiţii pot fi de mai multe feluri, ceea ce conduce la găsirea unor tipuri diferite de funcţii spline cubice (naturală, parabolică). Micula prezintă două cazuri pentru impunerea condiţiilor la limită suplimentare. Unul presupune ca funcţia spline să fie prelungită în afara intervalului [*a*, *b*] prin periodicitate cu perioada *b*-*a*. O altă posibilitate este de a impune condiţii derivatelor la capetele intervalului (în x_1 şi x_N). În fiecare caz se obţine câte un sistem de ecuaţii liniar univoc rezolvabil.

Sistemele liniare permit o rezolvare matriceală, care necesită un număr considerabil de calcule. Timpul de calcul poate fi destul de mare la implementarea acestor metode pe un sistem numeric de prelucrare.

În al doilea caz, în scrierea matricelor se obţin şi termenii de forma $D_i/6$. Aceştia reprezintă diferenţa divizată de ordinul doi a valorilor (y_{i-1}, y_i, y_{i+1}), pentru i=2, 3, ..., N-1. Este o observaţie ce va fi utilizată mai târziu în această teză.

Funcţiile spline de interpolare au proprietăţi de convergenţă remarcabile. Este de amintit faptul că sunt multe cazuri în care polinoamele de interpolare (Bernstein, Lagrange) sunt divergente. Proprietăţile de convergenţă sunt îmbunătăţite prin cerinţe mai tari de continuitate. Astfel, dacă funcţia spline $s \in C^4[a, b]$, ordinul de convergenţă se îmbunătăţeşte [60]. Este demonstrat şi faptul că ordinul de convergenţă nu creşte pentru clase şi mai netede de funcţii[60].

În prelucrarea numerică a semnalelor, pentru interpolarea şi aproximarea cu funcţii spline, iniţial se apela la rezolvarea matriceală, utilizând tehnici numerice tradiţionale. Însă, această abordare presupune calcularea inversei unei matrice şi o multitudine de operaţii care pot complica mult algoritmii de prelucrare. Problema interpolării poate fi rezolvată şi prin utilizarea funcţiilor B-spline şi a tehnicilor de filtrare numerică.

3.4. Funcţii B-spline

Spaţiul funcţiilor spline polinomiale de gradul n şi spaţiere unitară este notat S_1^n. Acest spaţiu este definit de I. J. Schoenberg astfel [15, 79, 81, 111]:

$$S_1^n = \left\{ s^n(x) = \sum_{k=-\infty}^{+\infty} c(k)\beta^n(x-k), x \in \Re, c \in l_2 \right\} \tag{3.20}$$

unde $\beta^n(x)$ este *funcţia B-spline simetrică de gradul n*:

$$\beta^n(x) = \sum_{j=0}^{n+1} \frac{(-1)^j}{n!} \binom{n+1}{j} \left(x + \frac{n+1}{2} - j\right)^n \mu\left(x + \frac{n+1}{2} - j\right), x \in \Re \tag{3.21}$$

$\mu(x)$ este funcţia treaptă unitate (3.22), iar termenul $\binom{n+1}{j}$ reprezintă coeficienţii binomiali daţi de relaţia (3.23) [108, 111].

$$\mu(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0. \end{cases} \tag{3.22}$$

$$\binom{n+1}{j} = \frac{(n+1)!}{(n+1-j)!\, j!} \tag{3.23}$$

I. J. Schoenberg a definit funcţiile B-spline cu ajutorul diferenţelor divizate sub denumirea de *funcţii spline cu deficienţă* sau *cu variaţie diminuată* (în engleză: *diminishing spline functions*) [65].

Se poate spune că funcţiile B-spline sunt de bază pentru construcţia funcţiilor spline, după cum se observă şi din denumirea acestora: B – de la „basic”

(în engleză: *basic*=de bază) [18]. Sunt foarte des utilizate deoarece sunt funcţii cu suport compact şi sunt funcţiile spline polinomiale cele mai scurte posibil [81, 108].

Ele pot fi reprezentate şi prin funcţii putere trunchiate [100, 110]:

$$\beta^n(x) = \sum_{j=0}^{n+1} \frac{(-1)^j}{n!} \binom{n+1}{j} \left(x + \frac{n+1}{2} - j\right)_+^n, \qquad x \in \Re \tag{3.24}$$

Relaţia (3.24) justifică afirmaţia lui Gh. Micula [60] că o funcţie spline cubică poate fi reprezentată cu ajutorul funcţiilor putere trunchiate.

Orice funcţie B-spline centrată de gradul *n* şi definită pe intervalul compact [-h(n+1)/2, h(n+1)/2] poate fi reprezentată astfel [32]:

$$\beta_h^n(x) = \sum_{j=0}^{n+1} \frac{(-1)^j}{n!} \binom{n+1}{j} \frac{\left(x + h(n+1)/2 - j\right)_+^n}{h^{n+1}}, \qquad x \in \Re \tag{3.25}$$

Datorită reprezentării prin (3.20), orice funcţie spline polinomială poate fi construită ca sumă ponderată de funcţii B-spline deplasate şi este caracterizată în mod unic prin secvenţa de coeficienţi B-spline *c*(*k*) [15]. În consecinţă, are structura unui semnal discret, chiar dacă reprezentarea modelului este continuă [100, 111].

Funcţia B-spline de gradul *n* poate fi construită recursiv, pe baza funcţiei B-spline de gradul *n*-1. Astfel, funcţia poate fi determinată prin convoluţia de (*n*+1) ori a impulsului dreptunghiular simetric β^0 (reprezentat în figura 3.2):

$$\beta^0(x) = \begin{cases} 1, & x \in \left[-\frac{1}{2}, \frac{1}{2}\right) \\ 0, & altfel \end{cases} \tag{3.26}$$

$$\beta^n(x) = \beta^0 * \beta^0 * \ldots * \beta^0(x) \qquad \text{(de } (n+1) \text{ ori)} \tag{3.27}$$

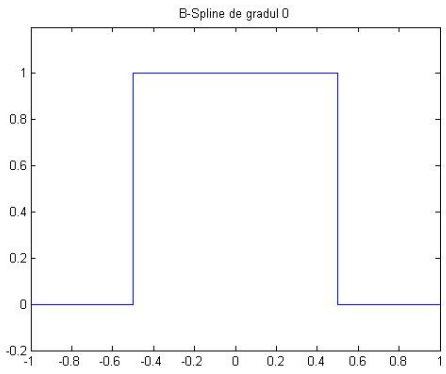


Fig.3.2. Impulsul dreptunghiular simetric β^0.

Funcţiile B-spline astfel definite sunt simetrice, iar pentru gradul *n* de la 0 la 3 sunt reprezentate în figura de mai jos:

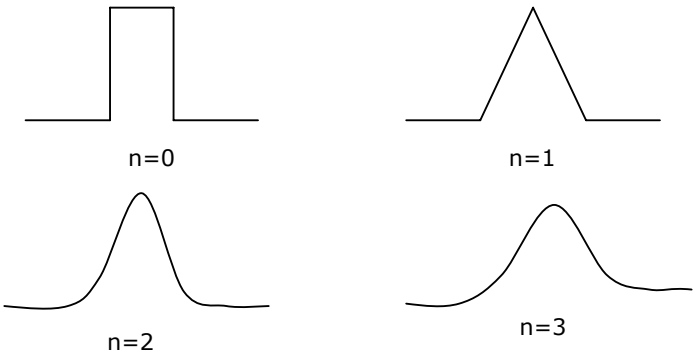


Fig.3.3. Funcţiile β^0, β^1, β^2 şi β^3.

Pentru n=0 (funcţia *spline constantă*) şi n=1 (*spline liniară*), valorile coeficienţilor sunt egale cu cele ale funcţiei în noduri [100].

Aceste funcţii prezintă şi avantajul de a fi uşor de manipulat, deoarece derivatele şi integralele lor se pot calcula prin formulele de recurenţă [83, 111]:

$$\frac{d\beta^n(x)}{dx} = \beta^{n-1}\left(x+\frac{1}{2}\right) - \beta^{n-1}\left(x-\frac{1}{2}\right) \tag{3.28}$$

$$\int_{-\infty}^{x} \beta^n(x)dx = \sum_{k=0}^{+\infty} \beta^{n+1}\left(x-\frac{1}{2}-k\right) \tag{3.29}$$

Prin derivarea funcţiei B-spline putem calcula derivata oricărei funcţii spline $s^n(x)$ reprezentată prin coeficienţii săi B-spline [111]:

$$\frac{\partial s^n(x)}{\partial x} = \sum_{k=-\infty}^{+\infty} c(k)\frac{\partial \beta^n(x-k)}{\partial x} \tag{3.30}$$

Se defineşte funcţia B-spline extinsă cu un factor m [108]:

$$\beta_m^n(x) = \beta^n(x/m) \tag{3.31}$$

Aceasta satisface proprietatea de convoluţie [98]:

$$\beta_m^n(x) = \frac{1}{m}\beta_m^{n-1} * \beta_m^0(x) = \frac{1}{m^n}\underbrace{\beta_m^0 * \ldots * \beta_m^0(x)}_{de\,(n+1)ori} \tag{3.32}$$

Funcţiile B-spline cubice (n=3) sunt folosite cel mai des în aplicaţii, în special pentru interpolări de bună calitate. Ele se pot reprezenta prin formula următoare [100]:

$$\beta^3(x) = \begin{cases} \frac{2}{3} - |x|^2 + \frac{|x|^3}{2}, & 0 \le |x| < 1 \\ \frac{(2-|x|)^3}{6}, & 1 \le |x| < 2 \\ 0, & 2 \le |x| \end{cases} \tag{3.33}$$

Reprezentarea grafică a funcţiei B-spline cubice se găseşte în figura 3.4.

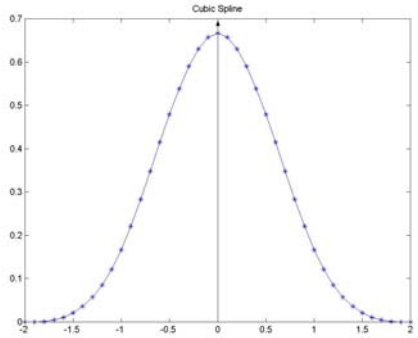
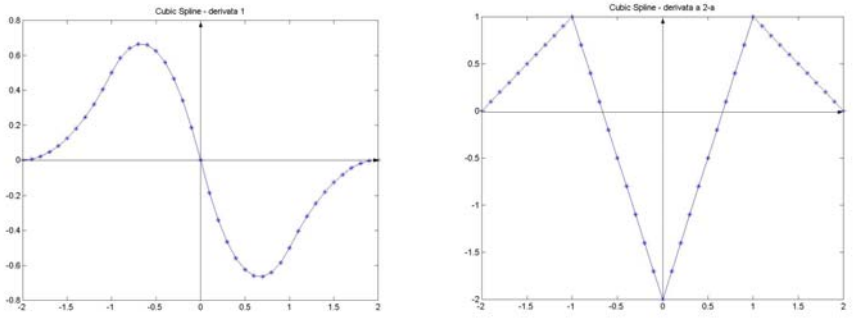


Fig.3.4. Funcţia B-spline cubică.



a) derivata întâi b) derivata a doua

Fig.3.5. Derivatele funcţiei B-spline cubică.

În figura 3.5 sunt prezentate derivatele întâi (3.5.a)) şi a doua (3.5.b)) ale funcţiei B-spline cubice. Acestea pot fi descrise prin următoarele formule:

$$\beta^{3'}(x)=\begin{cases}\frac{(2+x)^2}{2}, & -2\le x<-1\\ -2x-\frac{3x^2}{2}, & -1\le x<0\\ -2x+\frac{3x^2}{2}, & 0\le x<1\\ -\frac{(2-x)^2}{2}, & 1\le x<2\\ 0, & 2\le |x|\end{cases} \tag{3.34}$$

$$\beta^{3''}(x)=\begin{cases}2+x, & -2\le x<-1\\ -2-3x, & -1\le x<0\\ -2+3x, & 0\le x<1\\ 2-x, & 1\le x<2\\ 0, & 2\le |x|\end{cases} \tag{3.35}$$

Se pot defini funcţiile B-spline discrete (centrată şi deplasată) prin eşantionarea funcţiilor B-spline continue cu un factor de expansiune *m* [108, 111]. Funcţiile B-spline discrete centrate de gradul *n* sunt date prin:

$$b_m^n(k)=\beta^n(k/m) \tag{3.36}$$

Pentru factorul *m* impar, se defineşte funcţia spline discretă de gradul 0 prin relaţia 3.37. Se poate observa că aceasta este o fereastră dreptunghiulară de lăţime *m*, centrată în origine [111].

$$b_m^0=\begin{cases}1, & -m/2\le k\le m/2\\ 0, & altfel\end{cases} \tag{3.37}$$

Unul din marile avantaje ale utilizării funcţiilor B-spline în prelucrarea semnalelor este că oferă o trecere uşoară între domeniile continuu şi discret [96, 100, 108, 111]. Astfel, metode şi concepte matematice valabile în studiul funcţiilor continue pot fi utilizate pentru a dezvolta noi tehnici în prelucrarea numerică a semnalelor.

Un alt avantaj major derivă din faptul că sunt funcţii cu suport compact, definite pe intervale foarte scurte [18].

3.5. Tehnici de interpolare cu funcţii spline în prelucrarea numerică a semnalelor şi imaginilor

Până în anii ’90, interpolarea cu funcţii spline a fost puţin utilizată în prelucrarea semnalelor. Asta se întâmpla deoarece abordarea clasică a problemelor de interpolare şi aproximare cu funcţii spline polinomiale presupunea lucrul cu matrice, inversarea acestora şi un număr mare de calcule [100, 111]. S-au căutat

tehnici de calcul mai eficiente pentru determinarea şi implementarea unor algoritmi rapizi.

Problema interpolării a fost abordată în anii '70 prin prisma unor tehnici simple de filtrare numerică. În 1978, H. S. Hou şi H. C. Andrews [34] propun utilizarea unor filtre digitale pentru a realiza interpolarea unui semnal cu ajutorul funcţiilor B-spline cubice. Metoda presupune parcurgerea a două faze. În prima fază, din semnalul de intrare, se calculează un set de coeficienţi B-spline prin metoda matriceală. Aceşti coeficienţi sunt aduşi la intrarea unui filtru ce realizează interpolarea semnalului tot cu ajutorul funcţiilor B-spline cubice. Pentru filtru se propune atât o implementare software, cât şi una hardware. Implementarea hard se face printr-o schemă de filtre numerice ce presupune utilizarea mai multor circuite analogice simple, deci este uşor de realizat. Metoda de interpolare este utilizată cu succes pentru mărirea şi micşorarea unor imagini. În [34] sunt prezentate câteva exemple.

Însă, această abordare nu este foarte eficientă deoarece coeficienţii B-spline sunt determinaţi prin metoda clasică, metodă care presupune şi calcularea inversei unei matrice. Implementarea numerică a metodei este destul de greoaie şi presupune un număr mare de operaţii care trebuiesc efectuate [100].

Idea a fost reluată şi dezvoltată de M. Unser, A. Aldroubi şi M. Eden în anii '90 [107-112], în cadrul unui program al *National Institutes of Health* din Bethesda, USA. Din 1997, M. Unser şi colaboratorii săi de la *Biomedical Imaging Group,* din cadrul institutului *Ecole Polytechnique Federale* din Lausanne au extins cercetările şi asupra altor tipuri de funcţii spline [124]. Rezultatele au fost publicate de-a lungul timpului în mai multe lucrări (în reviste sau la conferinţe de specialitate), lucrări care pot fi consultate pe pagina de web a institutului [124]. Cercetările au aplicabilitate directă în prelucrarea numerică a imaginilor, cu precădere în prelucrarea imaginilor biomedicale [68].

Conceptele de eşantionare generalizată [41-43] şi interpolare generalizată [9, 96, 97] permit utilizarea unei plaje mai largi de funcţii de bază (de exemplu, funcţiile B-spline) [9, 10, 100]. Interpolarea generalizată presupune parcurgerea a două etape: prefiltrare şi sinteza semnalului. În etapa de prefiltrare, care este esenţială, se calculează coeficienţii modelului (respectiv coeficienţii B-spline). Aceştia sunt apoi utilizaţi pentru a realiza reconstrucţia sau interpolarea semnalului cu un anumit factor de interpolare [9].

În multe aplicaţii se preferă realizarea unei aproximări bune, în locul interpolării exacte [12]. Câştigul este dat de reducerea complexităţii calculelor şi uşurinţa de a realiza algoritmi rapizi de interpolare. Aceştia pot fi implementaţi pe sisteme numerice de calcul şi permit procesarea unui volum mare de date într-un timp mai scurt decât în cadrul metodelor clasice [99-112].

În lucrările [107-112], se propun mecanisme simple pentru proiectarea filtrelor ce evaluează transformatele B-spline directă şi indirectă. Sunt descrise noi tehnici de prelucrare care pot fi utilizate pentru diferenţierea semnalelor, filtrarea, netezirea spline şi aproximarea prin metoda celor mai mici pătrate.

Conceptul de filtrare B-spline presupune aplicarea unui operator de filtrare asupra reprezentării unui semnal prin funcţii B-spline continue [108, 111].

În continuare, va fi prezentat algoritmul de interpolare cu funcţii B-spline şi tehnici de filtrare numerică dezvoltat de M. Unser şi colaboratorii [100, 108, 111, 112], algoritm utilizat în prelucrarea imaginilor. Algoritmul este prezentat pentru prima dată în [108], lucrare care a fost publicată în 1991. În aceasta se specifică şi faptul că este pentru prima dată când transformata B-spline directă este

implementată prin filtrarea recursivă. Ulterior, algoritmul a fost reluat, completat şi modificat pentru diverse aplicaţii, toate din domeniul prelucrării imaginilor.

După cum s-a arătat în subcapitolul 3.4, orice funcţie spline $s(x)$ de gradul n poate fi caracterizată prin coeficienţii B-spline $c(k)$ şi funcţia B-spline $\beta^n(x)$ [15, 111]. Funcţia B-spline $\beta^n(x)$ este o funcţie continuă. Pentru a realiza şi implementa un algoritm numeric de interpolare avem nevoie de reprezentarea discretă a acesteia.

S-a definit funcţia B-spline discretă ca fiind obţinută prin eşantionarea funcţiei B-spline de ordin n $\beta^n(x)$, cu o rată de eşantionare m [100,111]:

$$b_m^n(k) = \beta^n(x/m)\Big|_{x=k} \tag{3.38}$$

Se dau eşantioanele semnalului de intrare $y(k)$ cu k=0, 1, ..., N-1 (N reprezintă numărul de eşantioane al semnalului ce urmează a fi prelucrat). Se consideră că $y(k)$ poate fi aproximat printr-o funcţie spline polinomială de gradul n $s(x)$. Aceasta este unic caracterizată prin coeficienţii săi B-spline $c(k)$.

$$s(x) = \sum_{l\in Z} c(l)\beta^n(x-l)\Big|_{x=k} = y(k) \tag{3.39}$$

Teoria interpolării impune ca funcţia să fie cu suport infinit. Pentru aceasta, semnalul iniţial se extinde prin oglindire în ambele părţi. Această practică este des utilizată în procesarea imaginilor. Prelungirea prin oglindire poate fi descrisă prin relaţiile [112]:

$$y(-k) = y(k+1), k = 0, \dots, N-1$$

$$y(k) = y(2N-k), k = N, \dots, 2N-1$$

Schematic, semnalul $y(k)$ poate fi interpolat cu un factor m prin intermediul unei transformări indirecte, [111]:

$$y_m(k) = y^n(x)\Big|_{x=k/m} = b_m^n * [c]_{\uparrow m}(k) = b_m^n * \left[\left(b_1^n\right)^{-1} * y\right]_{\uparrow m}(k) \tag{3.40}$$

Pas cu pas, procesul va fi descris în continuare. Pentru relaţia (3.38) se aplică transformata Z şi se obţine:

$$B_m^n(z) = \sum_{k\in Z} b_m^n(z) z^{-k} \tag{3.41}$$

Utilizând funcţia B-spline discretă, relaţia între eşantioanele de intrare şi coeficienţii B-spline se poate scrie prin prisma convoluţiei a două semnale:

$$y(k) = \left(b_1^n * c\right)(k) \tag{3.42}$$

Dacă se aplică transformata Z în formula (3.42), obţinem:

$$Y(z) = C(z)B_1^n(z) \tag{3.43}$$

Se defineşte operatorul invers convoluţiei:

$$\left(b_1^n\right)^{-1}(k) \xleftrightarrow{Z} 1 / B_1^n(z) \tag{3.44}$$

Pasul crucial în interpolarea cu funcţii B-spline este determinarea coeficienţilor astfel încât funcţia să ia valorile secvenţei discrete $y(k)$ [108]. Astfel, pentru coeficienţii B-spline se poate scrie următoarea relaţie:

$$c(k) = \left(b_1^n\right)^{-1} * y(k) \tag{3.45}$$

$$\Rightarrow \qquad C(z) = \left[B_1^n(z)\right]^{-1} Y(z) \tag{3.46}$$

Deci, coeficienţii B-spline se obţin din eşantioanele cunoscute prin aplicarea unui filtru numeric, funcţia de transfer a filtrului fiind $\left[B_1^n(z)\right]^{-1}$. Acesta este denumit „filtru B-spline direct" [100, 108, 112] sau „transformata B-spline directă". Este un filtru simetric, filtru cu răspuns infinit la impuls (IIR) şi trebuie ales cu grijă, astfel încât să fie un filtru stabil [100]. Funcţia de transfer a filtrului poate fi descrisă prin relaţia [108, 111, 112]:

$$\left[B_1^n(z)\right]^{-1} = \frac{z^{\lceil n/2 \rceil}}{b_1^n(\lceil n/2 \rceil)\prod_{i=1}^{\lceil n/2 \rceil}(z - z_i)\left(z - z_i^{-1}\right)} \tag{3.47}$$

unde $\{(z_i, z_i^{-1}): |z_i| \le 1, i=1, \ldots, [n/2]\}$ sunt polii filtrului, care pot fi grupaţi în pereche datorită faptului că filtrul este simetric.

Filtrul poate fi descompus în filtre simetrice exponenţiale elementare conectate în cascadă. La rândul lor, acestea pot fi separate în două filtre complementare recursive de ordinul 1: unul cauzal şi celălalt anticauzal.

În consecinţă, coeficienţii se calculează prin intermediul unor ecuaţii recursive simple, pentru care se impun condiţii iniţiale care derivă din prelungirea semnalului prin oglindire [112].

Acest pas de determinarea a coeficienţilor B-spline mai este descris şi ca *prefiltrare*, făcându-se analogie cu teoria eşantionării a lui Shannon, unde se recomandă operaţia de prefiltrare cu un filtru anti-aliere [100].

Pentru a realiza interpolarea semnalului cu un factor m, şirul coeficienţilor B-spline $c(k)$ este supraeşantionat $c_m(k)$ şi adus la intrarea unui „filtru B-spline indirect" care are funcţia de transfer $B_m^n(z)$.

Supraeşantionarea şirului de coeficienţi cu un factor întreg m [100], este descrisă prin relaţia (3.48).

$$c_m(k) = [c]_{\uparrow m}(k) = \begin{cases} c(i) & \text{pentru } k = mi; \\ 0 & \text{în rest.} \end{cases} \tag{3.48}$$

Eşantioanele semnalului obţinut la sfârşitul procesului de interpolare se vor nota în continuare cu $y_i(k)$.

$$Y_i(z) = C_m(z) B_m^n(z) \tag{3.49}$$

Operaţia este denumită şi „transformată B-spline indirectă”. Termenul $B_m^n(z)$ reprezintă un filtru simetric cu răspuns finit la impuls (*FIR*). Şi acest filtru, dacă are ordin mai mare, poate fi implementat prin conectarea în cascadă a unor filtre de ordinul 1 sau 2 [100, 111].

Semnalul iniţial se poate reconstitui prin aplicarea filtrului $B_1^n(z)$ asupra coeficienţilor B-spline. Semnalul reconstituit va fi notat $y_r(k)$.

Întreg algoritmul de calcul al coeficienţilor, reconstrucţia semnalului şi interpolarea cu un factor m, este redat schematic în figura 3.6 [100, 111, 123].

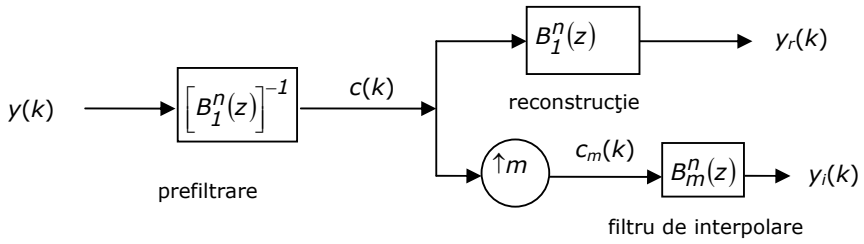


Fig.3.6. Algoritmul de interpolare B-spline.

Algoritmul este prezentat pentru cazul general. Poate fi particularizat pentru orice valori ale gradului n al funcţiei B-spline şi factorului de interpolare m. La creşterea gradului n se pot obţine rezultate mai bune pentru procesul de interpolare, dar în acelaşi timp creşte şi volumul de calcule.

Avantajul major al metodei este că se utilizează filtre digitale, care sunt simplu de realizat [100, 108, 111, 112]. Atât coeficienţii, cât şi eşantioanele semnalului interpolat, sunt calculaţi prin operaţii simple de adunare şi înmulţire. În comparaţie cu metodele matriceale clasice, sunt necesare un număr redus de calcule, de complexitate mică. Acestea pot fi uşor transpuse într-un program. Se creează astfel un algoritm de calcul stabil numeric, mai rapid şi mai uşor de implementat decât pentru oricare alte tehnici numerice. Procesul poate fi rulat cu uşurinţă pe un sistem numeric de calcul (calculator, procesor de semnal). Eficienţa rezultă şi din faptul că lucrăm cu un număr minim de termeni, deoarece pentru a calcula valoarea în orice punct este nevoie de mai puţini termeni decât în cazul oricăror alte funcţii. Se poate spune că funcţiile B-spline sunt funcţiile de bază cu cel

mai scurt suport [100, 111]. De aceea, ele sunt ideale pentru a fi folosite în prelucrarea imaginilor.

În ultimele 3 decenii s-au dezvoltat multe aplicaţii de interpolare spline în domeniul procesării numerice a imaginilor. Unele studii din acest domeniu [100, 111, 113, 124], propun utilizarea şi a altor tipuri de funcţii spline. În formula (3.49), filtrul $B_m^n(z)$ poate fi înlocuit cu alte filtre spline, în funcţie de condiţiile impuse de problemă. Câteva exemple de astfel de filtre sunt: filtru spline de netezire (în engleză: *smooting spline*), filtrul B-spline obţinut cu metoda celor mai mici pătrate (în engleză: *least squares splines*), filtrul spline cardinal, filtre cu fereastră Kaiser, filtre trunchiate [100, 113, 123, 124]. Au fost propuse soluţii optimizate şi pentru cazurile în care semnalele sunt însoţite de zgomot [73]. În continuare, vor fi prezentate câteva dintre soluţiile descrise în literatura de specialitate.

Interpolarea cu *funcţii spline cardinale* este des întâlnită în prelucrarea imaginilor [100, 110, 111, 114]. Funcţia de interpolare poate fi scrisă în raport cu valorile sale discrete astfel:

$$s^n(x) = \sum_{k=-\infty}^{+\infty} s(k)\eta^n(x-k) \tag{3.50}$$

$$\eta^n(x) = \sum_{k=-\infty}^{+\infty} \left(b_1^n\right)^{-1}(k)\beta^n(x-k) \tag{3.51}$$

În (3.50) şi (3.51), $\eta^n(x)$ este *funcţia spline cardinală de gradul n* şi reprezintă răspunsul la impuls în timp continuu al interpolatorului spline polinomial. Funcţia mai este cunoscută şi sub denumirea de *funcţie spline fundamentală de gradul n*. Cea mai importantă proprietate a sa, din punctul de vedere al interpolării, este că funcţia este zero în toate nodurile, mai puţin în origine [111].

Relaţia (3.50) reprezintă formula de interpolare în care se folosesc valorile semnalului ca şi coeficienţi. Procesul este posibil deoarece $\eta^n(x)$ are aceleaşi proprietăţi de interpolare ca şi funcţia sinus cardinal *sinc*(x) [100]. Cu cât se creşte gradul funcţiei n, cu atât caracteristica acesteia este mai apropiată de caracteristica funcţiei *sinc*(x) [3, 111]. Funcţia *sinc*(x) corespunde cazului ideal de interpolare pentru semnale de bandă limitată. Însă, pentru $n \geq 2$ aceste funcţii nu mai au suport compact, ceea ce constituie un dezavantaj [100, 111]. Este mult mai eficientă reprezentarea prin funcţii B-spline decât prin funcţii spline cardinale.

În [111] se afirmă că în cazul funcţiilor B-spline, spre deosebire de cele cardinale, la creşterea gradului *n,* acestea tind să semene mai mult cu clopotul lui Gauss. Deşi, în alte lucrări anterioare [107-110], dar şi ulterioare [100, 105], aceeaşi autori afirmă că şi funcţiile B-spline tind spre funcţia *sinc*(x).

Funcţiile spline cardinale exponenţiale sunt utilizate pentru definirea unor operatori care realizează trecerea de la reprezentarea în timp continuu, la cea în timp discret [105].

Pentru cazurile în care semnalul este însoţit de zgomot, este mai potrivit ca pentru aproximare să fie utilizate *funcţiilor spline de netezire* (*smooting splines*) [100, 111, 112, 114]. Acestea se caracterizează printr-un parametru λ. Parametrul poate fi ales în funcţie de 2 cerinţe, care sunt oarecum în contradictoriu:

- aproximarea să fie cât mai apropiată de valorile datelor iniţiale;

- funcţia să fie suficient de netedă.

I. J. Schoenberg a fost cel care a propus pentru prima dată utilizarea funcţiilor spline de netezire [111]. El a considerat cazul general în care nodurile nu sunt egal distanţate. În problemele de filtrare numerică prezentate mai sus se utilizează cazul particular al eşantioanelor egal distanţate.

În această situaţie, coeficienţii B-spline se obţin la ieşirea unui *filtru spline de netezire*. Funcţia de transfer pentru filtrul de ordinul 1 este dată de [112]:

$$S_{\lambda}^{1}(z)=\frac{1}{1+\lambda\left(-z^{-1}+2-z\right)} \tag{3.52}$$

Utilizarea filtrelor de netezire de ordin mai mare este ceva mai complicată deoarece polii filtrului depind de λ şi nu pot fi calculaţi cu uşurinţă în orice caz [112].

O altă soluţie este interpolarea cu *funcţii spline prin metoda celor mai mici pătrate* (în engleză: *least squares splines*). Carl deBoor a propus acest tip de funcţii pentru a reduce gradele de libertate ale funcţiei de aproximare. El a prezentat soluţia generală utilizând metoda celor mai mici pătrate [16, 111].

Prin prisma filtrării numerice, această variantă presupune calcularea unui număr mai mic de coeficienţi B-spline. Se propune parcurgerea următorilor paşi pentru determinarea coeficienţilor [108, 110-112]:

- operaţia de prefiltrare cu un filtru care are funcţia de transfer B_m^n;
- decimarea cu un factor m;
- postfiltrare cu filtrul recursiv, care are funcţia de transfer dată de relaţia (3.53):

$$S_m^n(z)=\frac{1}{\frac{1}{m}\sum_{k=0}^{m-1}B_m^n\left(\left[ze^{j2\pi k}\right]^{1/m}\right)^2} \tag{3.53}$$

Procesul de aproximare a semnalelor cu acest tip de funcţii este redat schematic în figura 3.7 [111].

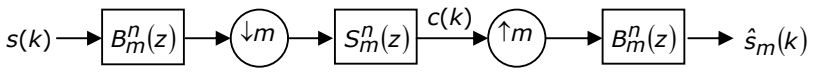


Fig.3.7 Aproximarea cu filtre spline prin metoda celor mai mici pătrate.

Filtrele descrise în acest subcapitol sunt prezentate în [112] pentru mai multe valori ale parametrilor n şi m. Sunt redate explicit funcţiile de transfer ale filtrelor B-spline direct pentru valori ale lui n de la 0 la 7 şi ale filtrelor prin metoda celor mai mici pătrate de grad 0, 1, 2 şi 3 pentru cazurile în care m ia valorile 2 şi 3. Metodele de interpolare de mai sus sunt folosite pentru prelucrarea unor imagini şi sunt prezentate câteva rezultate. Este evaluată şi metoda clasică, matriceală, de interpolare spline cubică. Aceasta necesită efectuarea unui număr mai mare de operaţii şi folosirea de memorie suplimentară pentru stocarea rezultatelor

intermediare, în comparaţie cu oricare dintre metodele ce utilizează filtrarea numerică. Se argumentează, prin exemple, faptul că abordarea clasică are o complexitate mai mare, necesită mai mult timp de calcul şi resurse suplimentare.

Teoria funcţiilor B-spline polinomiale poate fi generalizată şi în cazul în care funcţia are gradul *n* număr fracţionar. Se definesc *funcţiile spline fracţionare* pe baza aceleiaşi relaţii (3.21), unde *n* este număr fracţionar [33, 100, 104, 116]. Aceste funcţii păstrează toate proprietăţile funcţiilor B-spline, însă nu mai au suport compact [100]. Ele permit lucrul facil cu derivate fracţionare, deoarece derivarea fracţionară este transpusă în domeniul B-spline prin diferenţe finite. În [116] sunt prezentate aceste funcţii împreună cu posibilitatea de a fi folosite în tomografia computerizată.

De asemenea, se definesc şi *funcţiile B-spline complexe* [25]. Pentru acestea, gradul funcţiei este un număr complex. Ele păstrează proprietăţile funcţiilor B-spline, mai puţin cea de suport compact.

Pentru aproximări de calitate, se propune utilizarea unei clase de funcţii numite *funcţii de ordin maxim şi suport minim* MOMS (în engleză: maximal-order-minimal-support) [8, 10]. Acestea sunt funcţii polinomiale pe subintervale adiacente, cu noduri unitar spaţiate, şi reprezintă o combinaţie liniară de funcţii B-spline şi derivate ale lor. Generic, pot fi descrise prin relaţia [96, 97]:

$$MOMS^n(x) = \beta^n(x) + \sum_{k=1}^{n} c_k \frac{d^k}{dx^k} \beta^n(x) \tag{3.54}$$

În [10] se prezintă mai multe variante de funcţii MOMS şi modul în care acestea pot fi construite. Sunt descrise funcţiile MOMS optime (O-MOMS), sub-optime (SO-MOMS) şi de interpolare (I-MOMS). Funcţia de interpolare I-MOMS de ordinul 4 (gradul 3) este dată de:

$$\varphi_I^3(x) = \beta^3(x) - \frac{1}{6} \frac{d^2}{dx^2} \beta^3(x) \tag{3.55}$$

Prin experimente ce presupun rotirea succesivă de mai multe ori a unor imagini se arată că această nouă clasă de funcţii bază MOMS este superioară funcţiilor B-spline de acelaşi grad [8, 10]. De asemenea, se demonstrează că pentru pasul suplimentar de prefiltrare costurile de implementare sunt aproape neglijabile în comparaţie cu cele rezultate în pasul de interpolare. Pentru cel de-al doilea pas, costurile sunt date în mod direct de mărimea intervalului pe care este definită funcţia bază $\varphi(x)$. De aceea, funcţia bază trebuie să aibă un suport cât mai scurt posibil pentru a eficientiza algoritmii de prelucrare [10].

Au fost elaborate şi metode de interpolare liniară care, prin adăugarea pasului de prefiltrare, oferă rezultate mai bune în unele aplicaţii de prelucrare a imaginilor [11].

Metodele de interpolare prezentate pot fi extinse şi în spaţii multidimensionale. Filtrarea directă B-spline poate fi aplicată succesiv pe fiecare dimensiune, după coordonate, pentru a obţine coeficienţii B-spline. La fel se poate face şi cu filtrarea indirectă pentru a realiza interpolarea datelor. De exemplu, în problemele de prelucrare a imaginilor, operaţiile se efectuează succesiv pe fiecare linie şi coloană [100, 111]. Utilizând funcţii B-spline liniare pot fi create modele numerice 2D pentru algoritmii de reconstrucţie a imaginilor tomografice [5].

Pentru toate metodele de prelucrare cu funcţii spline prezentate în lucrările lui M. Unser şi ale colaboratorilor săi [124] sunt date doar aplicaţii din domeniul procesării imaginilor. Funcţiile spline, alături de funcţiile wavelet, sunt preferate în unele aplicaţii din prelucrarea imaginilor medicale, aplicaţii în care este necesară reprezentarea datelor sub forma unui model continuu mai degrabă, decât discret [100, 116]. Pentru un semnal discret se poate obţine uşor reprezentarea acestuia printr-o funcţie spline. În unele dintre aceste lucrări se fac referiri la posibile aplicaţii pentru semnale unidimensionale, precum interpolare, derivare, filtrare, reducerea zgomotului şi compresia datelor [100, 111]. Se precizează că unele operaţii precum diferenţiere sau integrare ar putea fi mai uşor de efectuat în domeniul transformatei B-spline. Dar, niciunde nu sunt prezentate şi rezultate ale aplicării acestor algoritmi în cazul unor semnale des utilizate în prelucrarea semnalelor, ci doar pentru prelucrarea imaginilor.

Metodele dezvoltate şi publicate de M. Unser şi colaboratorii în [2-3, 8-10, 100, 107-112] între anii 1991-2001 au servit ca bază teoretică şi pentru alţi cercetători în dezvoltarea unor aplicaţii în prelucrarea imaginilor. Au fost comparate metodele de interpolare pentru diverse valori ale gradului *n*, sau folosind variate tipuri de funcţii spline. De asemenea, s-au făcut modificări asupra filtrelor spline, obţinând noi metode de interpolare. Toate acestea sunt expuse şi evaluate prin exemple în care sunt prelucrate imagini. În continuare, vor fi prezentate câteva dintre aceste lucrări cu rezultate şi concluzii considerate semnificative.

În [46] se face o comparaţie între mai multe metode de interpolare folosite în prelucrarea imaginilor prin prisma complexităţii calculelor, a timpului de execuţie şi a erorilor obţinute pentru câteva exemple des întâlnite în practică. Sunt comparate următoarele metode de interpolare: sinc trunchiat, cu funcţii în scară, liniară, pătratică, B-spline cubică, cu polinoame Lagrange şi Gaussiană. Din punct de vedere al similitudinii cu imaginea originală, metoda de interpolare cu funcţii B-spline oferă unul dintre cele mai bune rezultate. În ceea ce priveşte timpul de execuţie, algoritmul este cel mai rapid. Se precizează că la creşterea gradului funcţiei B-spline se poate îmbunătăţi calitatea interpolării, dar creşte şi complexitatea calculelor.

În lucrarea [47], care vine ca o completare la [46], se face o analiză amănunţită a interpolării cu funcţii B-spline de gradul 2, 4, şi 5. Pentru fiecare sunt date expresiile explicite ale filtrelor cu răspuns infinit la impuls şi sunt analizate rezultatele în cazul interpolării unor imagini. În continuare sunt prezentate câteva concluzii ale studiului:

- Cu cât creşte gradul funcţiei de interpolare, cu atât răspunsul în frecvenţă al sistemului tinde să fie tot mai aproape de cel al unui sistem ideal de interpolare. Această concluzie mai apare şi în alte studii [100, 108, 111].
- Metodele care utilizează funcţii B-spline de gradul 4 şi 5 oferă o calitate mai bună a interpolării, comparativ şi cu interpolarea liniară, deoarece produc cel mai mic număr de pixeli eronaţi.
- Odată cu creşterea gradului funcţiei de interpolare, erorile se micşorează semnificativ, dar creşte numărul de operaţii necesare.

Comparaţii între diverse tehnici de interpolare utilizate în aplicaţii de prelucrare a imaginilor se mai regăsesc şi în alte lucrări [89, 96, 97].

B. Vrcelj şi P.P. Vaiddyanathan fac observaţia că filtrul B-spline direct din algoritmul de interpolare cu funcţii B-spline dezvoltat de M. Unser şi colaboratorii nu poate fi folosit pentru interpolarea în timp real a semnalelor unidimensionale [123]. Ei propun mai multe variante de trunchiere a răspunsului în frecvenţă al acestui

filtru, astfel încât coeficienţii B-spline să fie obţinuţi la ieşirea unor filtre cu răspuns finit la impuls. În general, filtrele FIR pot fi utilizate pentru interpolarea în timp real a semnalelor.

În [123] sunt prezentate 3 variante de filtre B-spline cubice trunchiate:

- 5TFIR – filtru trunchiat la 5 componente (five-tap)
- 7TFIR – filtru trunchiat la 7 componente (seven-tap)
- 5KFIR – filtru trunchiat la 5 componente cu fereastră Kaiser.

Aceste filtre, împreună cu filtrul B-spline direct sunt utilizate pentru interpolarea cu un factor m=3 a 11 imagini cu caracteristici diferite. Rezultatele arată că înlocuirea filtrului IIR cu aceste filtrele FIR mai scurte, se poate face fără a compromite rezultatul interpolării, diferenţele fiind foarte mici. Metoda a fost testată şi pentru interpolare spline de ordin mai mare. Concluzia a fost că pentru gradul n=5 este suficient să se folosească un filtru cu trunchiere la 7 componente, iar pentru n=7, unul cu trunchiere la 9 componente.

Autorii [123] propun şi o nouă abordare a filtrului B-spline indirect. Numărul de operaţii efectuate la interpolare ar putea fi redus, deoarece filtrul indirect se aplică unui semnal care are multe valori egale cu zero. Aceste valori rezultă prin supraeşantionarea şirului de coeficienţi cu un factor m. Se propune implementarea filtrului prin conectarea în paralel, a unor filtre care reprezintă componentele polifazice ale filtrului B-spline indirect. Filtrele se aplică direct coeficienţilor calculaţi şi apoi se face supraeşantionarea pe fiecare ramură. Schema oferă complexitate redusă şi interpolarea este reversibilă. Un alt avantaj al metodei este că aceste filtre pot opera în paralel, la o frecvenţă redusă.

Într-o lucrare din 2012, se propun noi variante de realizare a pasului de prefiltrare [76]. Ştiind că întreg setul de date de intrare este cunoscut de la început, se propun noi modalităţi de implementare în program a celor 2 filtre: cauzal şi anti-cauzal. Prin acestea se urmăreşte micşorarea volumului de memorie şi a timpului de execuţie necesare pentru calculul coeficienţilor B-spline.

Se pot folosi unele funcţii B-spline modificate pentru interpolarea şi reeşantionarea imaginilor [28, 29, 30]. Funcţiile B-spline modificate sunt obţinute ca o combinaţie liniară de funcţii B-spline deplasate care au grade diferite.

$$\beta^{mod}(x) = \sum_{m=0}^{M} \sum_{n \in Z} l_{mn} \beta^{m}(x-n) \tag{3.56}$$

Fiecare funcţie B-spline are o pondere diferită l_{mn}. Prin optimizarea acestor parametri se pot obţine funcţii de interpolare cu caracteristici îmbunătăţite, fără a creşte complexitatea calculelor. În exemplele prezentate de autori, prin alegerea atentă a ponderilor, interpolarea şi reconstrucţia imaginilor cu ajutorul funcţiilor B-spline modificate oferă o calitate vizuală mai bună (conturul este mai bine conservat) [28].

Alături de metodele de interpolare deja cunoscute, în [30] se prezintă 2 variante de combinare a funcţiilor B-spline: combinaţia *funcţii B-spline de gradul trei-şi-unu* (sau combinaţia de gradul 3), şi combinaţia *funcţii B-spline de gradul cinci-trei-şi-unu* (sau combinaţia de gradul 5). Se preferă combinarea funcţiilor de grad impar deoarece acestea au nodurile în punctele de interpolare. Sunt date formele explicite ale acestor funcţii, reprezentarea lor, modul de optimizare a parametrilor şi metodele de implementare eficientă a acestora prin filtre. S-au realizat operaţii succesive de rotire a anumitor imagini utilizând funcţiile prezentate

în lucrare. Rezultatele arată că funcţiile B-spline modificate propuse de autori, oferă rezultate mult mai bune decît celelalte metode în cazul imaginilor ce conţin frecvenţe înalte.

Pornind de la modul de construcţie al funcţiilor MOMS în spaţiul unidimensional, s-au dezvoltat noi familii de funcţii spline pentru aplicaţii în spaţiile bidimensionale: hex-spline şi box-spline [14, 121]. Funcţiile hex-spline sunt funcţii spline al căror suport este de formă hexagonală. Funcţiile box-spline sunt funcţii spline definite pe spaţii multidimensionale. Cu ajutorul acestora se determină noi metode de interpolare a imaginilor eşantionate hexagonal.

În studii mai recente sunt prezentate şi alte aplicaţii cu funcţii B-spline din domeniul prelucrării imaginilor:

- metode de calcul rapid al *funcţiilor B-spline poliarmonice* [4];
- metode de filtrare cu funcţii B-spline de diverse grade utilizate pentru segmentarea imaginilor [6];
- modalităţi de legătură între funcţiile spline şi fractali [115].

3.6. Alte aplicaţii cu funcţii spline

Modurile de abordare a problemei interpolării au ca principală limitare faptul că nu sunt proiectate pentru a minimiza pierderea informaţiei, ci pentru a minimiza eroarea de interpolare [63]. Pentru mărirea imaginilor rezultatele sunt bune, însă pot să apară probleme în aplicaţii de micşorare a imaginilor. Astfel, pentru redimensionarea imaginilor, s-au dezvoltat tehnici de prelucrare cu funcţii spline care utilizează şi operaţiile de derivate, respectiv integrare a acestor funcţii. Algoritmii de prelucrare se pot aplica atât pentru date eşantionate uniform, cât şi pentru cele obţinute prin eşantionare neuniformă. Plecând de la metodele de redimensionare a imaginilor care utilizează interpolare cu funcţii spline, în [63] se descrie un procedeu de redimensionare a imaginilor în care sunt utilizate diferenţele finite şi diferenţele divizate ale funcţiilor B-spline.

Integrarea numerică (procesul invers derivării) este definită cu ajutorul diferenţelor divizate ale funcţiilor B-spline. Este apoi utilizată pentru a face trecerea de la coordonate neuniforme la coordonate uniforme [62]. Algoritmul presupune parcurgerea următorilor paşi:

- interpolarea datelor cu ajutorul funcţiilor spline neuniforme;
- integrarea numerică;
- reeşantionarea uniformă;
- filtrarea numerică: cu un filtru corespunzător obţinerii diferenţelor finite centrate şi apoi postfiltrare cu ajutorul filtrului IIR propus de Unser [111, 112].

Metoda este eficientă şi complexitatea sa este independentă de spaţierea nodurilor semnalului de intrare [63].

Funcţiile B-spline pot fi utilizate cu succes şi în probleme de conversie numeric-analogică. Un exemplu este generatorul de semnal cu funcţii spline pătratice dezvoltat de M. Kamada şi colaboratorii [40]. Unul dintre circuitele propuse a fost folosit pentru realizarea unor echipamente audio de înaltă fidelitate [100]. M. Kamada, împreună cu colaboratorii de la Universitatea Ibaraki din Hitachi, Japonia, au dezvoltat mai multe studii pe această temă.

Pornind de la faptul că funcţiile B-spline continue se pot scrie ca o convoluţie de impulsuri dreptunghiulare β^0, se demonstrează că interpolarea prin funcţii în scară reprezintă o aproximare a funcţiilor B-spline discrete, atunci când perioada de

eşantionare tinde la zero [36, 37]. Funcţia B-spline de gradul 0 are forma unui impuls dreptunghiular şi pe baza ei se construiesc funcţii spline de grad superior. Metoda presupune generarea unor impulsuri dreptunghiulare deplasate, care apoi sunt integrate, inversate şi adunate succesiv, în mai mulţi paşi, pentru a obţine la ieşire o funcţie B-spline de un anumit grad şi variantele sale deplasate. Astfel, generatorul de semnal propus de Kamada în 1995 se poate realiza printr-o combinaţie de circuite analogice simple [40].

În 1998, circuitul analogic este descris printr-un filtru digital rapid şi utilizat pentru generarea funcţiilor B-spline [37]. Semnalul obţinut este apoi convertit în semnal analogic şi poate fi utilizat şi pentru generarea altor tipuri de impulsuri.

Ulterior, în 2005 metoda a fost îmbunătăţită pentru a realiza un generator de impulsuri de bandă ultra-largă [49].

Subiectul este de actualitate, fiind reluat şi în 2010 de o echipă de cercetători de la Brest [1]. Aceştia propun optimizarea metodei de generare a impulsurilor de bandă ultra-largă pentru o anumită aplicaţie.

V. Burov şi colaboratorii filtrează datele obţinute experimental prin intermediul unor filtre de interpolare (IFIR), realizate pe baza funcţiilor B-spline de aproximare [13]. Soluţia prezentată de aceştia este matriceală.

În 2009, firma Boeing patentează un filtru de interpolare spline cubică (IFIR) destinat aplicaţiilor de prelucrare numerică a semnalelor de mare viteză [131]. Acest filtru prezintă avantajul că are mai puţine componente decât filtrele clasice şi poate fi implementat utilizând tehnologia VLSI (Very Large Scale Integration). În 2012, este patentat un sistem digital de comunicaţie radio în componenţa căruia apare acest filtru [132].

3.7. Algoritmul rapid de interpolare spline cubică (algoritmul Unser)

3.7.1. Prezentarea algoritmului

Interpolarea cu funcţii B-spline cubice, împreună cu tehnicile de filtrare numerică, sunt foarte des folosite în rezolvarea problemelor de prelucrare numerică a imaginilor. Aceasta reprezintă o soluţie viabilă, rapidă, ce necesită timp de calcul mic şi complexitate redusă, oferind o bună aproximare. Este folosit cu succes pentru lucrul cu un volum mare de date, aşa cum este cazul reprezentării imaginilor.

În continuare, se analizează algoritmul rapid de interpolare spline cubică descris de M. Unser în [100, 112]. Acesta reprezintă o particularizare a algoritmului general de interpolare cu funcţii B-spline prezentat în subcapitolul anterior.

Funcţia B-spline cubică este o funcţie B-spline de gradul 3 (n=3). Filtrul B-spline direct $\left[B_1^3(z)\right]^{-1}$ este reprezentat prin funcţia de transfer [100]:

$$\left[B_1^3(z)\right]^{-1} = \frac{6}{z + 4 + z^{-1}} \tag{3.57}$$

Fie semnalul de intrare $y(k)$ cu k = 0, 1, ..., N-1. Algoritmul presupune, în primul rând, calculul coeficienţilor $c(k)$ ai modelului B-spline prin aplicarea filtrului

B-spline direct $\left[B_1^3(z)\right]^{-1}$ asupra eşantioanelor semnalului de intrare. Acest filtru se descompune în 2 filtre: unul cauzal şi unul anticauzal (conform relaţiei (3.58)), care sunt conectate în cascadă.

$$\left[B_1^3(z)\right]^{-1} = \frac{6}{z+4+z^{-1}} = 6\left(\frac{1}{1-z_1z^{-1}}\right)\left(\frac{-z_1}{1-z_1z}\right) \tag{3.58}$$

Coeficienţii se obţin la ieşirea sistemului prezentat în figura 3.8 [100].

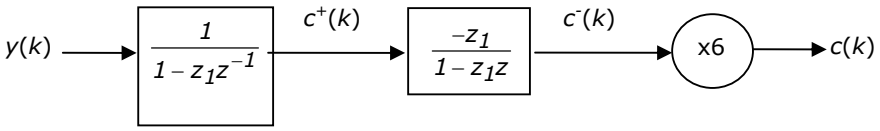


Fig.3.8. Determinarea coeficienţilor B-spline.

Determinarea valorilor acestor coeficienţi se face printr-un algoritm recursiv care presupune calculul factorilor $c^+(k)$ şi $c^-(k)$:

$$c^+(k) = y(k) + z_1\, c^+(k\text{-}1), \qquad k = 1, \ldots, N\text{-}1 \tag{3.59}$$

$$c^-(k) = z_1\,(c^-(k+1) - c^+(k)), \qquad k = N\text{-}2, \ldots, 0 \tag{3.60}$$

unde: $$z_1 = -2 + \sqrt{3} = -0.2679 \tag{3.61}$$

şi $$c(k) = 6c^-(k) \tag{3.62}$$

Pentru acest proces trebuie stabilite condiţiile iniţiale. Condiţia de la care se porneşte este ca eşantioanele semnalului de intrare să poată fi reconstituite exact din coeficienţii $c(k)$ calculaţi. Semnalul iniţial se prelungeşte la capete prin oglindire, astfel încât:

$$y(k)=y(l), \text{ pentru } (k+l)\bmod(2N\text{-}2)=0 \tag{3.63}$$

Semnalul rezultat este un semnal periodic de perioadă $2N$-2 . Pentru prima formulă, aceea de calcul al factorilor $c^+(k)$, iniţializarea se face cu [100]:

$$c^+(0) = \sum_{k=0}^{+\infty} y(k) z_1^k \tag{3.64}$$

Aceasta nu se poate utiliza în practică deoarece semnalul de intrare are un număr finit de termeni. Dacă semnalul se prelungeşte la capete prin oglindire, relaţia (3.64) se poate scrie:

$$c^+(0) = \frac{1}{1 - z_1^{2N-2}} \sum_{k=0}^{2N-3} y(k) z_1^k \tag{3.65}$$

Implementarea ultimei formule poate necesita un volum crescut de calcule, ceea ce conduce la complicarea algoritmului şi necesitatea unor timpi mari de execuţie. Unser propune formula (3.66), cu factorul k_0 dat de (3.67), unde ε este nivelul de acurateţe dorit:

$$c^+(0) = \sum_{k=0}^{k_0} y(k) z_1^k \tag{3.66}$$

$$k_0 > \log \varepsilon / \log |z_1| \tag{3.67}$$

Cel de-al doilea proces recursiv se iniţializează cu:

$$c^-(N-1) = \frac{z_1}{\left(z_1^2 - 1\right)} \left(c^+(N-1) + z_1 c^+(N-2) \right) \tag{3.68}$$

Acest proces de determinare a coeficienţilor din eşantioanele de intrare se mai numeşte şi prefiltrare. Având coeficienţii, se poate trece la a doua etapă a algoritmului: calcularea valorilor interpolate.

Pentru factorul m=1 se obţine reconstrucţia semnalului iniţial. Şirul de coeficienţi este adus la intrarea filtrului care are funcţia de transfer $B_1^3(z)$.

$$B_1^3(z) = \frac{z + 4 + z^{-1}}{6} \tag{3.69}$$

Acestuia îi corespunde:

$$b_1^3(k) = \left\{ \frac{1}{6}, \frac{2}{3}, \frac{1}{6} \right\} \tag{3.70}$$

Pentru interpolarea cu un factor m>1 şirul coeficienţilor B-spline este supraeşantionat conform formulei (3.48) şi adus la intrarea filtrului $B_m^3(z)$. Expresia filtrului B-spline indirect este dată de formula (3.71), pentru un factor de interpolare m=2 [99].

$$B_2^3(z) = \frac{32 + 23\left[z + z^{-1}\right] + 8\left[z^2 + z^{-2}\right] + \left[z^3 + z^{-3}\right]}{48} = \\ = \frac{2}{3} + \frac{23}{48}\left[z + z^{-1}\right] + \frac{1}{6}\left[z^2 + z^{-2}\right] + \frac{1}{48}\left[z^3 + z^{-3}\right] \tag{3.71}$$

Corespunzător:

$$b_2^3(k) = \left\{ \frac{1}{48}, \frac{1}{6}, \frac{23}{48}, \frac{2}{3}, \frac{23}{48}, \frac{1}{6}, \frac{1}{48} \right\} \tag{3.72}$$

În lucrările studiate, acest algoritm şi alte metode bazate pe acesta au fost folosite pentru a efectua operaţii de prelucrare a imaginilor [100, 112, 123]. În continuare, vor fi prezentate câteva rezultate proprii obţinute prin aplicarea algoritmului asupra unor semnale bine cunoscute.

3.7.2. Rezultate experimentale proprii

Într-un studiu propriu, algoritmul rapid de interpolare spline cubică a fost implementat prin programe *Matlab* şi aplicat mai multor categorii de semnale binecunoscute, ca: sinus, cosinus, dreaptă, semnal treaptă, semnal triunghiular [91]. Coeficienţii B-spline, eşantioanele semnalului reconstituit (pentru m=1) şi interpolat (factor de interpolare m=2), au fost calculate în fiecare caz în parte. S-au utilizat funcţiile B-spline cubice (n=3), filtrul B-spline direct (aşa numita „transformată B-spline directă"), fiind cel din formula (3.58). Pentru reconstrucţia semnalelor s-a utilizat filtrul B-spline indirect cu funcţia de transfer $B_1^3(z)$ dată de relaţia (3.69), iar pentru interpolarea cu factor m=2 s-a folosit filtrul $B_2^3(z)$ dat de (3.71).

Schema de interpolare particularizată pentru n=3 şi m=2 este redată în figura următoare.

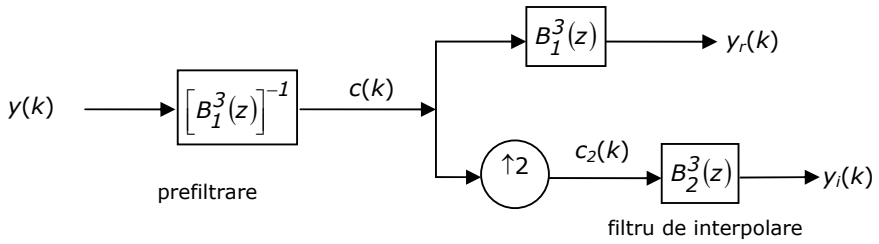


Fig.3.9. Algoritmul rapid de interpolare spline cubică pentru m=2.

Algoritmul a fost aplicat mai multor tipuri de semnale, precum: semnale sinus şi cosinus, un semnal care are doar componentă continuă (o dreaptă), semnal treaptă, semnal triunghiular, semnal în V, un semnal electrocardiogramă. Pentru semnalele periodice, s-au luat în considerare mai multe situaţii: un număr de 12, 50, 100 sau 120 eşantioane pe perioadă, corespunzătoare unor frecvenţe de eşantionare diferite. Numărul de eşantioane pe perioadă s-a notat cu M. Numărul eşantioanelor N ce reprezintă şirul $y(k)$ a fost ales astfel încât să corespundă la 1, 3 sau 5 perioade din semnalul de intrare. Rezultatele obţinute au permis formularea unor concluzii şi observaţii. Acestea au dus la căutarea unor metode noi de determinare a coeficienţilor şi elaborarea unor algoritmi îmbunătăţiţi de interpolare.

Din eşantioanele semnalului de intrare $y(k)$ se determină şirul de coeficienţi $c(k)$. Operaţia de iniţializare a procesului recursiv de calcul al coeficienţilor B-spline necesită stabilirea valorii parametrului k_0 din relaţia (3.66). Vom alege trei valori pentru nivelul de acurateţe ε şi, din (3.62) calculăm k_0. În urma aplicării algoritmului, rezultatele vor fi analizate pentru a vedea influenţa acestui factor. Se aleg:

$$\varepsilon = 0{,}001 \Rightarrow k_0 > 5{,}2452;\ k_0 = 6$$
$$\varepsilon = 0{,}0001 \Rightarrow k_0 > 6{,}9936;\ k_0 = 7$$
$$\varepsilon = 0{,}00001 \Rightarrow k_0 > 8{,}7421;\ k_0 = 9$$

Se vor urmări modificările care apar în calculul coeficienţilor la schimbarea valorii lui k_0. Astfel se studiază influenţa condiţiilor iniţiale asupra valorilor interpolate şi a erorilor de interpolare.

Pentru exemplificare, s-au luat ca semnale de intrare eşantioanele corespunzătoare lui $y(k)=sin(2\pi k/M)$ şi $y(k)=cos(2\pi k/M)$. Pentru M s-au ales 2 valori: 12 şi 120. Acestea corespund eşantionării semnalelor $sin(t)$ şi $cos(t)$ cu o frecvenţă de eşantionare mică şi respectiv, una de 10 ori mai mare.

Reprezentarea datelor în *Matlab* s-a făcut cu dublă precizie: reprezentare în virgulă mobilă pe 64 de biţi (tipul *double*). Toate valorile obţinute mai mici de 10^{-8} au fost considerate neglijabile. În practică, valori mult mai mari decât acestea sunt considerate erori acceptabile pentru multe aplicaţii. Semnalele de intrare folosite (atât $sin(t)$, $cos(t)$, cât şi altele care vor fi prezentate în continuare) se consideră scalate, astfel încât valorile acestora se regăsesc doar în intervalul [-1; 1].

Fiecare set de N eşantioane a fost adus la intrarea sistemului din figura 3.9. Pentru fiecare din cele 3 valori ale lui k_0 (6, 7 şi 9) s-au calculat coeficienţii $c(k)$, şirurile $y_r(k)$ şi $y_i(k)$. Algoritmul de calcul al coeficienţilor $c(k)$, al semnalului reconstruit $y_r(k)$ şi al semnalului interpolat cu m=2 $y_i(k)$ a fost implementat în *Matlab* şi se regăseşte în *Anexa 1*. Rezultatele obţinute sunt prezentate parţial în *Anexa 1*.

În tabelul 3.1 sunt redate câteva rezultate semnificative. Sunt prezentate valorile primilor 3 şi a ultimilor 2 coeficienţi B-spline ai şirului pentru fiecare caz în parte.

Tabelul 3.1. Coeficienţi $c(k)$ pentru diverse valori k_0

$y(k)$	M	k_0	$c(0)$	$c(1)$	$c(2)$	$c(M-1)$	$c(M)$
$sin(2\pi k/M)$	12	6	-0,302225	0,604353	0,884809	-0,604338	0,302169
		7	-0,302139	0,604330	0,884815	-0,604338	0,302169
		9	-0,302167	0,604338	0,884813	-0,604338	0,302169
	120	6	-0,030182	0,060447	0,102409	-0,060459	0,030229
		7	-0,030244	0,060463	0,102404	-0,060459	0,030229
		9	-0,030231	0,060460	0,102405	-0,060459	0,030229
$cos(2\pi k/M)$	12	6	1,046618	0,906542	0,523363	0,906508	1,046745
		7	1,046767	0,906502	0,523374	0,906508	1,046745
		9	1,046744	0,906508	0,523372	0,906508	1,046745
	120	6	1,000584	0,999051	0,994985	0,999085	1,000457
		7	1,000423	0,999094	0,994974	0,999085	1,000457
		9	1,000454	0,999086	0,994976	0,999085	1,000457

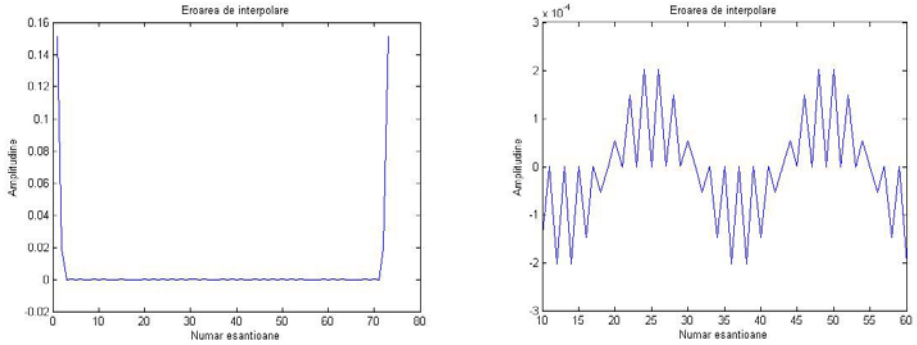
Analizând rezultatele obţinute la modificarea valorii lui k_0, se pot face următoarele observaţii:

- Pentru acelaşi semnal de intrare există diferenţe între valorile primilor coeficienţi.
- Numărul de coeficienţi care se modifică depinde de pasul de eşantionare.
- Diferenţele dintre valorile coeficienţilor devin neglijabile la un moment dat, după care şirurile rămân identice la schimbarea valorii lui k_0.
- Ultimele valori corespunzătoare fiecărui şir sunt identice.
- Valorile coeficienţilor urmăresc variaţia eşantioanelor de intrare şi sunt apropiate de valorile acestora.

Calculul primilor coeficienţi depinde în mod direct de valoarea parametrului k_0. De aici apar diferenţele semnificative de la începutul şirurilor. Procesul recursiv are ca efect reducerea diferenţelor pe măsură ce se înaintează în şir.

Pentru aceleaşi semnale de intrare s-au comparat şi erorile de interpolare pentru cazul în care factorul de interpolare este m=2. Câteva fişiere cu date reprezentative se regăsesc în *Anexa 1*. Se observă că valorile acestor erori nu depăşesc un anumit prag, cu excepţia celor de la capetele şirului. Acestea vor fi denumite în continuare „erori la capete”. O analiză a acestor erori va fi prezentată după ce vor fi discutate şi rezultatele pentru alte tipuri de semnale de intrare.

În figura 3.10 sunt prezentate erorile de interpolare în cazul semnalului $y(k)=cos(2\pi k/M)$, cu M=12, k=0, 1,..., N şi N=36. În acest exemplu s-a ales k_0=7. În figura 3.10.a) este redat întregul şir de valori obţinute. Se observă că la capete erorile ating valori de ordinul 10^{-1}. În figura 3.10.b), din şir au fost îndepărtate erorile la capete şi se vede că erorile de interpolare au valori de maxim $2\cdot10^{-4}$.



a) pentru toate eşantioanele prelucrate b) fără erorile la capete

Fig.3.10. Eroarea de interpolare pentru $cos(2\pi k/12)$, k_0=7.

Numărul eşantioanelor afectate de erori semnificative de la începutul şirului creşte odată cu scăderea lui k_0. S-ar putea spune că o valoare cât mai mare a lui k_0 ar duce la rezultate mai bune. Aceste erori, de la începutul şirului, se datorează şi faptului că relaţia de iniţializare (3.66) este o particularizare a lui (3.64), unde $k_0\to\infty$. Alegerea lui k_0 mai mare duce la mărirea numărului de termeni în formula (3.66), deci creşte volumul de calcule. Acesta poate fi privit ca o optimizare, însă, în multe cazuri, rezultatele obţinute nu pot justifica timpul de calcul suplimentar.

Pentru semnalul sinus s-au calculat erorile de interpolare pentru mai multe valori ale factorului M (numărul de eşantioane pe perioadă), de exemplu: 5, 6, 8, 10, 12, ...,240, în fiecare caz luând k_0=7. Numărul de eşantioane afectate de erorile la capete tinde să ajungă la jumătate din M. Cu cât k_0 devine mai mic în comparaţie cu M, cu atât mai mult creşte numărul de eşantioane afectate de erorile la capete.

Algoritmul a fost aplicat şi pentru cazul în care eşantioanele de intrare corespund unui semnal constant egal cu valoarea 1 (o dreaptă). S-a considerat un număr de 51 de eşantioane (N=51). Astfel, s-a ales:

$$y(k)=1, \text{ pentru } k=0, 1, 2,..., 50$$

În tabelul 3.2. sunt prezentate valorile coeficienţilor în câteva puncte k semnificative pentru cazul în care k_0=7. Pe ultima coloană sunt date valorile obţinute în urma interpolării semnalului de intrare cu un factor m=2.

Tabelul 3.2. Semnal $y(k)=1$, $k_0=7$

k	$c(k)$	$y_i(k/m)$
0	0,9996	0,8333
		0,9791
1	1	1
		1
2	0,9999	1
		0,9999
3	1	1
		1
4	0,9999	1
...	...	...
30	1	1
		1
31	1	1
...	...	...
49	1	1
		0,9791
50	1	0,8333

Se observă că primele valori pentru coeficienţi sunt diferite de 1, dar converg rapid spre 1, ajungând să fie egale cu valorile semnalului de intrare. În punctele de interpolare semnalul ia valori foarte apropiate de 1. Erorile de interpolare sunt mici, devenind neglijabile odată cu creşterea numărului de valori calculate [91]. Excepţie fac eşantioanele de la începutul şi sfârşitul şirului, apărând şi aici erori la capete.

Se pune în evidenţă încă o dată faptul că algoritmul este convergent datorită proprietăţilor funcţiilor spline polinomiale [100, 112]. Însă, unele mici oscilaţii apărute în şirul coeficienţilor duc la apariţia de oscilaţii şi în semnalul interpolat. Acesta poate fi un inconvenient, mai ales dacă avem de prelucrat un număr mic de eşantioane.

În practică, iniţializarea algoritmului recursiv se face în k=0 şi cu un număr finit de eşantioane k_0. Presupunem că semnalul iniţial este extins prin oglindire în ambele părţi, cu un număr suficient de mare de eşantioane şi apoi interpolat.

Atunci, în zona de mijloc a semnalului de la ieşire vom avea rezultate neinfluenţate de erorile la capete. Această zonă este cea în care au fost definite valorile iniţiale şi pentru care rezultatele sunt de interes. În acest caz, erorile de capete nu mai prezintă nici o importanţă [91].

Pentru semnalul $y(k)=1$, formula de iniţializare la stânga (3.64) poate fi scrisă astfel:

$$c^{+}(0)=\sum_{k=0}^{+\infty}y(k)z_1^k=\sum_{k=0}^{+\infty}z_1^k=\frac{1}{1-z_1} \tag{3.70}$$

Acum se pot evalua coeficienţii şi semnalul interpolat pentru cazul ideal în care parametrul k_0 tinde la ∞ (corespunzător unui semnal de intrare nemărginit). În tabelul 3.3. sunt prezentate câteva valori pentru coeficienţi şi semnalul interpolat cu factor $m=2$.

Tabelul 3.3. Semnal $y(k)=1$, $k_0=+\infty$

k	$c(k)$	$y_i(k/m)$
0	1	0,8333
		0,9791
1	1	1
		1
2	1	1
		1
3	1	1
		1
4	1	1
…	…	…
30	1	1
		1
31	1	1
…	…	…
49	1	1
		0,9791
50	1	0,8333

În această situaţie coeficienţii sunt egali cu datele de intrare, iar valorile semnalului interpolat sunt identice cu cele ale semnalului iniţial (rezultat aşteptat în cazul ideal). Pentru semnalul interpolat apar erori la capete şi în acest caz.

Pe baza rezultatelor de mai sus s-a stabilit valoarea $k_0=7$ pentru simulările următoare.

În figura 3.11 sunt prezentate atât eşantioanele de intrare cât şi coeficienţii calculaţi pentru cazul $y(k)=cos(2\pi k/12)$ unde $k=0, 1, ..., 36$ (o secvenţă de 3 perioade din $cos(t)$, cu 12 eşantioane pe perioadă).

Din rezultatele obţinute se poate vedea că valorile coeficienţilor urmăresc variaţia semnalului de la intrare şi sunt apropiate de valorile acestuia [91]. Acest lucru este valabil pentru toate semnalele pe care s-a testat algoritmul. Coeficienţii sunt egali cu eşantioanele în cazul în care se utilizează funcţii spline cardinale în procesul de determinare a şirului $c(k)$ [100, 112]. Dezavantajul acestui tip de funcţii

este că nu au suport compact, precum funcţiile B-spline polinomiale. Astfel se pierde avantajul de a putea calcula orice valoare interpolată utilizând maxim (n+1) funcţii bază.

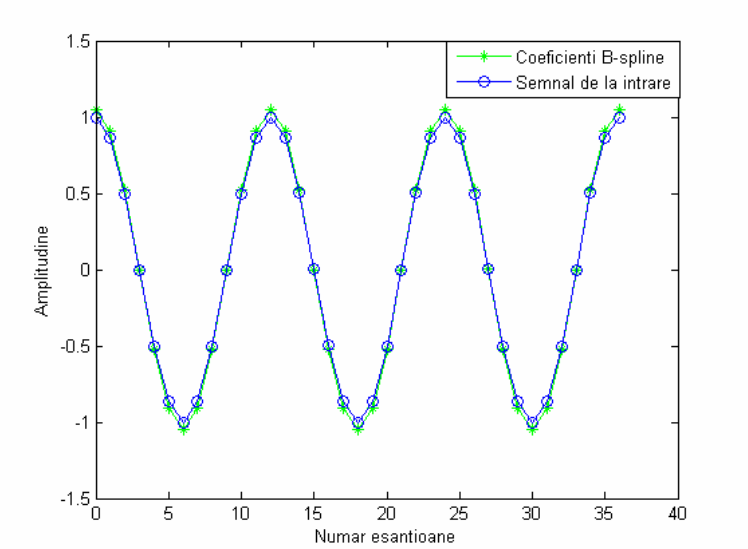


Fig.3.11. Eşantioanele de intrare şi coeficienţii determinaţi cu algoritmul Unser.

Pentru semnalele analogice $sin(t)$ şi $cos(t)$ am luat şiruri de eşantioane corespunzătoare mai multor frecvenţe de eşantionare. Astfel, avem cazuri în care diferă numărul de eşantioane pe perioadă (diverse valori pentru factorul M). Am analizat şi situaţiile în care la intrare avem eşantioane care corespund unei perioade, respectiv a trei şi cinci perioade din semnal.

Fie $y(k)=sin(2\pi k/M)$ cu k=0, 1, ..., N-1. Se prezintă în continuare situaţiile în care M=12, N=37, respectiv M=120, N=361. Acestea sunt cazurile în care secvenţele de intrare au 12 eşantioane pe perioadă, respectiv 120, şi conţin fiecare câte trei perioade din semnal. Câteva dintre valorile coeficienţilor $c(k)$ sunt comparate în tabelul 3.4.

Tabelul 3.4. Coeficienţi pentru $y(k)= sin(2\pi k/M)$, k_0=7

α	$sin(\alpha)$	M	Prima perioadă		A doua perioadă		A treia perioadă	
			k	$c(k)$	k	$c(k)$	k	$c(k)$
0	0	12	0	-0,302139	(M+0)	0	(2M+0)	0
		120	0	-0,030244	(M+0)	0	(2M+0)	0
π/6	0,5	12	1	0,604330	(M+1)	0,523372	(2M+1)	0,523372
		120	10	0,500228	(M+10)	0,500228	(2M+10)	0,500228
π/3	0,866025	12	2	0,884815	(M+2)	0,906508	(2M+2)	0,906509
		120	20	0,866421	(M+20)	0,866421	(2M+20)	0,866421

Acestea au fost obţinute în aceleaşi puncte α de pe caracteristica semnalului de intrare. În tabel sunt prezentate şi valorile funcţiei sinus în aceleaşi puncte.

În tabelul 3.5 sunt prezentate rezultatele pentru semnalele $y(k)=cos(2\pi k/M)$, unde k=0, 1, ..., N-1, pentru aceleaşi valori M=12, N=37, respectiv M=120, N=361.

Tabelul 3.5. Coeficienţi pentru $y(k)= cos(2\pi k/M)$, k_0=7

α	$cos(\alpha)$	M	Prima perioadă		A doua perioadă		A treia perioadă	
			k	$c(k)$	k	$c(k)$	k	$c(k)$
0	1	12	0	1,046767	(M+0)	1,046745	(2M+0)	1,046745
		120	0	1,000423	(M+0)	1,000457	(2M+0)	1,000457
π/6	0,866025	12	1	0,906502	(M+1)	0,906508	(2M+1)	0,906508
		120	10	0,866421	(M+10)	0,866421	(2M+10)	0,866421
π/3	0,5	12	2	0,523374	(M+2)	0,523372	(2M+2)	0,523372
		120	20	0,500228	(M+20)	0,500228	(2M+20)	0,500228

Valorile coeficienţilor diferă pentru şiruri de eşantioane obţinute din acelaşi semnal analogic la frecvenţe de eşantionare diferite. Valorile $y(k)$ şi $c(k)$ sunt apropiate atunci când semnalele de intrare au un număr mai mare de eşantioane pe perioadă (M=120), diferenţele fiind de ordinul 10^{-4}. Aceste diferenţe se măresc odată cu micşorarea frecvenţei de eşantionare. Pentru un pas de eşantionare de 10 ori mai mare (M=12), diferenţele ajung la valori de ordinul 10^{-2}, chiar 10^{-1} în cazul semnalului sinusoidal [91].

La începutul şirului de coeficienţi, diferenţele dintre aceştia şi valorile datelor de intrare sunt mai mari decât în rest. Acest fapt se datorează condiţiilor iniţiale impuse pentru determinarea lui $c^+(0)$. Pentru iniţializare se recurge la prelungirea prin oglindire a semnalului de intrare. Dacă datele de intrare corespund funcţiei cosinus (semnal continuu, par), în urma prelungirii se obţine tot un semnal continuu. Pentru sinus (care este funcţie impară), în urma prelungirii apar discontinuităţi în noul semnal. Acest tip de prelungire nu este avantajoasă în cazul semnalelor impare deoarece, prin oglindire, pentru semnalul rezultat apar discontinuităţi în punctele faţă de care s-a făcut oglindirea.

Pentru un semnal compus din mai multe perioade se obţine un şir de coeficienţi care prezintă o anumită periodicitate. Coeficienţii din a doua perioadă sunt diferiţi de cei din prima până la un punct, dar sunt egali cu cei din perioadele următoare. Valorile corespunzătoare fiecărei perioade sunt aproximativ egale, excepţie făcând primele şi ultimele din şir (acolo unde se face iniţializarea lui $c^+(0)$ şi $c^-(N\text{-}1)$). Din tabelele 3.4 şi 3.5 se poate observa diferenţa mare dintre valorile pentru $c(0)$ şi $c(M)$ (corespunzătoare primelor două perioade). Coeficienţii $c(M)$ şi $c(2M)$ (ce corespund perioadelor a doua şi a treia), au valori egale. În cazul semnalelor periodice cu M eşantioane pe perioadă, pentru coeficienţi, se poate scrie:

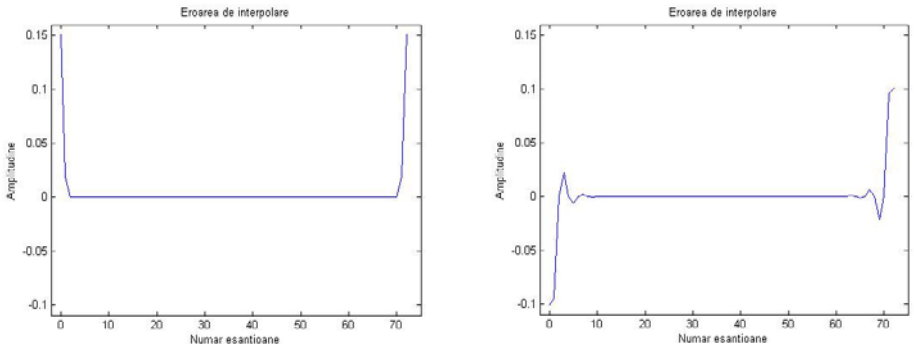
$$c(M+i)= c(2M+i)=\ldots= c((j\text{-}1)M+i) \text{ pentru } i=0,1\ldots, M\text{-}1 \qquad (3.71)$$

$$c(0) \approx c(M) \quad \text{şi} \quad c(1) \approx c(M+1)$$

Pentru aceleaşi date de intrare s-au analizat şi valorile interpolate cu un factor m=2. Erorile de interpolare sunt de ordinul 10^{-8} maximum, pentru M=120 şi 10^{-4}, pentru M=12. Excepţie fac un număr de eşantioane de la capetele şirurilor,

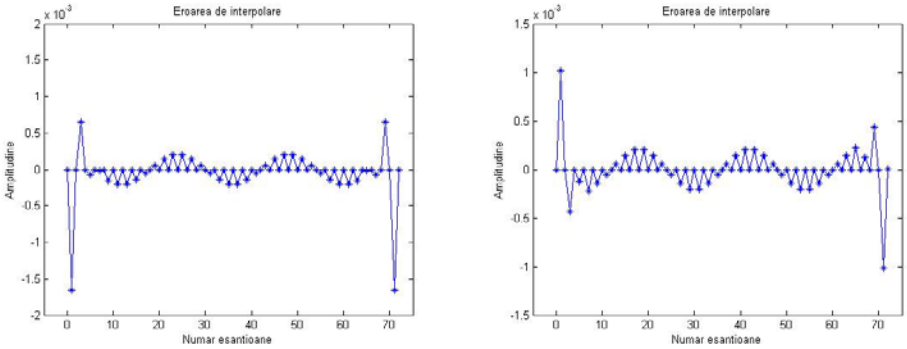
unde erorile sunt considerabil mai mari. Primele valori interpolate au erori de ordinul 10^{-1}. În figura 3.12 sunt prezentate erorile de interpolare obţinute în cazul semnalelor *cos*($2\pi k/M$) (3.12.a) şi *sin*($2\pi k/M$) (3.12.b) pentru M=12.

În *Matlab* există funcţii predefinite [66, 128] care realizează interpolarea unui şir de date. Aceleaşi semnale de mai sus au fost interpolate utilizând funcţia *interp1()*, precizând *spline* ca metodă de interpolare. Rezultatele obţinute pentru erorile de interpolare sunt prezentate în figura 3.13. Şi în aceste cazuri apar erori mai mari la capete. Pentru restul şirului, erorile de interpolare sunt mai mari decât în cazul interpolării cu algoritmul Unser.



a) cos($2\pi k/12$) b) sin($2\pi k/12$)

Fig.3.12. Erori la interpolarea cu algoritmul Unser.



a) cos($2\pi k/12$) b) sin($2\pi k/12$)

Fig.3.13. Erori la interpolarea cu funcţia predefinită din Matlab.

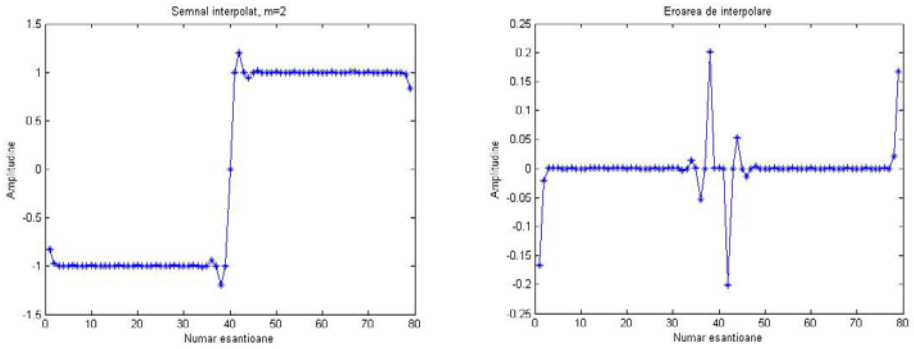
Aceste erori de interpolare, denumite „erori la capete”, au fost observate în toate cazurile studiate. Apariţia lor are la bază 2 cauze:

- Condiţiile iniţiale pentru determinarea coeficienţilor B-spline sunt date de un număr finit de eşantioane.
- Pentru calcularea fiecărui eşantion din semnalul interpolat se utilizează valori anterioare şi posterioare ale coeficienţilor. La capetele intervalului, unele dintre aceste valori sunt considerate 0 (zero). De exemplu, în punctul k=0 nu există c(-1) sau c(-2) pentru

a calcula valoarea $y_i(0)$ a semnalului interpolat. Pentru aceşti coeficienţi se consideră valoarea zero.

Până acum au fost discutate doar cazuri în care eşantioanele de intrare provin din semnale descrise prin funcţii continue şi cu derivatele continue. Acelaşi algoritm a fost aplicat şi unor semnale care prezintă discontinuităţi: semnal triunghiular şi semnal treaptă. S-a studiat şi cazul interpolării unui semnal electrocardiogramă (*ECG*).

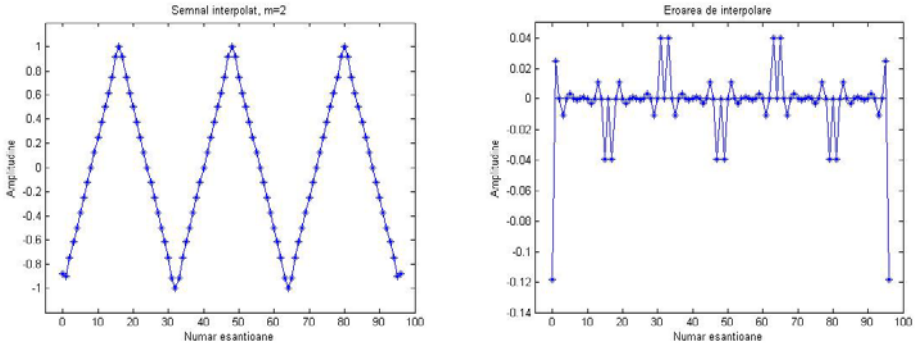
Câteva rezultate sunt prezentate în continuare. În figura 3.14 sunt redate un semnal treaptă interpolat cu un factor m=2 (în 3.14.a) şi eroarea de interpolare (în 3.14.b). Primele 20 de eşantioane ale semnalului de la intrare au valoarea „-1”, după care semnalul sarela valoarea „1” pentru următoarele 20 de eşantioane. La mijlocul semnalului apare o discontinuitate. În jurul acesteia, la interpolare apar erori semnificative care afectează un număr mic de valori (3 la stânga şi 3 la dreapta).



a) semnal interpolat cu m=2 b) eroarea de interpolare

Fig.3.14. Rezultate pentru semnal treaptă.

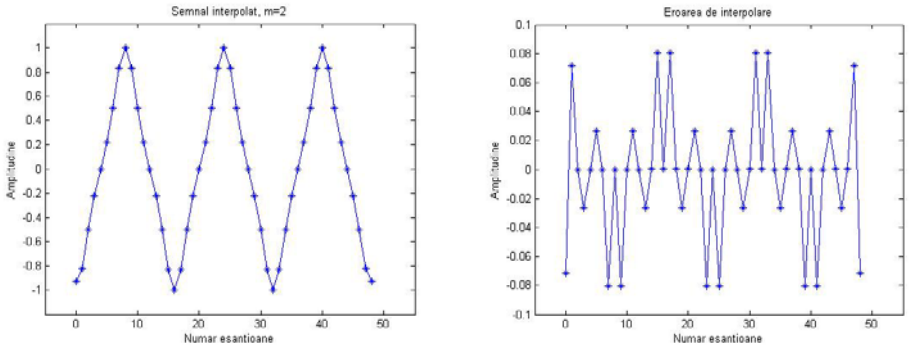
Cazul unui semnal triunghiular cu 16 eşantioane pe perioadă este redat în figura 3.15. Şi aici se observă valori mai mari pentru erorile de interpolare din jurul discontinuităţilor.



a) semnal interpolat cu m=2 b) eroarea de interpolare

Fig.3.15. Rezultate pentru semnal triunghiular cu 16 eşantioane pe perioadă.

Pentru o frecvenţă de eşantionare de 2 ori mai mică (triunghi cu 8 eşantioane pe perioadă), rezultatele sunt prezentate în figura 3.16. Erorile de interpolare din vecinătatea punctelor de discontinuitate cresc, însă cele de la capete scad. Erorile la capete sunt mai mici deoarece s-a considerat $k_0=7\cong M$. Aceasta poate fi considerată valoare mare în comparaţie cu numărul de eşantioane pe perioadă.

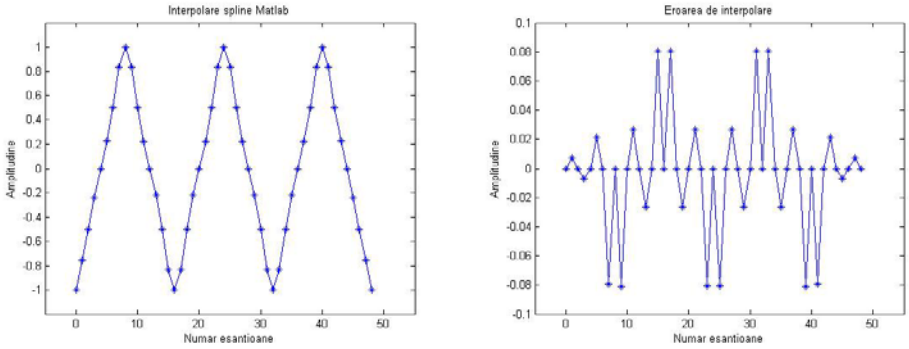


a) semnal interpolat cu $m=2$ b) eroarea de interpolare

Fig.3.16. Rezultate pentru semnal triunghiular cu 8 eşantioane pe perioadă.

Observaţiile anterioare referitoare la coeficienţi şi erorile de interpolare sunt valabile şi pentru semnalele cu discontinuităţi. Apar erori de interpolare suplimentare în zonele unde sunt punctele de discontinuite deoarece semnalele sunt aproximate prin funcţii spline cubice (3.39).

Acelaşi semnal a fost interpolat utilizând funcţia *interp1()* predefinită în *Matlab*, folosind metoda de interpolare *spline* (figura 3.17). Exceptând erorile la capete, erorile din vecinătatea punctelor de discontinuitate au valori aproximativ egale cu cele obţinute prin interpolare cu algoritmul Unser.



a) semnal interpolat cu $m=2$ b) eroarea de interpolare

Fig.3.17. Semnal triunghiular interpolat spline cu funcţia predefinită în Matlab.

Acelaşi lucru se întâmplă şi pentru semnalul treaptă la interpolarea cu funcţia *interp1()* din *Matlab*.

Pentru toate cazurile studiate, în noduri, erorile de interpolare au valori nesemnificative, deci funcţia de interpolare ia valorile semnalului de intrare (îndeplineşte condiţia (3.39)).

Algoritmul Unser a fost utilizat şi pentru reconstrucţia unui semnal sinus cu 120 de eşantioane pe perioadă care are suprapus un zgomot. Semnalul este reconstruit cu erori neglijabile.

S-a utilizat pentru studiu şi un semnal electrocardiogramă (ECG), obţinut la o frecvenţă de eşantionare de 200 Hz [74], având aproximativ 160 de eşantioane pe perioadă, nescalat. Semnalul reconstruit este prezentat în figura 3.18. Erorile de reconstrucţie sunt neglijabile. Reamintim că orice valoare mai mică de 10^{-8} a fost considerată zero şi că în *Matlab* s-a lucrat cu date de tip *double*.

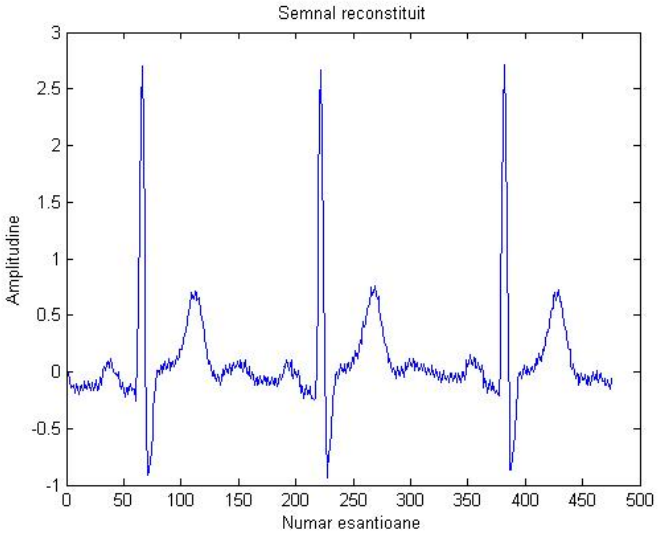


Fig.3.18. Semnal ECG reconstruit cu algoritmul Unser.

După cum s-a observat, eroarea de interpolare scade odată cu creşterea frecvenţei cu care a fost eşantionat semnalul de intrare (când eşantioanele sunt mai dese). S-a studiat modul în care variază valoarea maximă a erorii de interpolare (exceptând erorile la capete), la creşterea frecvenţei de eşantionare. Pentru semnalul $sin(2\pi k/M)$ s-au calculat erorile de interpolare pentru mai multe valori ale factorului M (5, 6, 8, 10, 12, ...,240). În figura 3.19 este prezentată variaţia erorii maxime de interpolare cu factor m=2 în funcţie de numărul de eşantioane pe perioadă (raportul frecvenţă de eşantionare f_e/frecvenţă semnal f_s) pentru acest semnal. Valorile scad logaritmic.

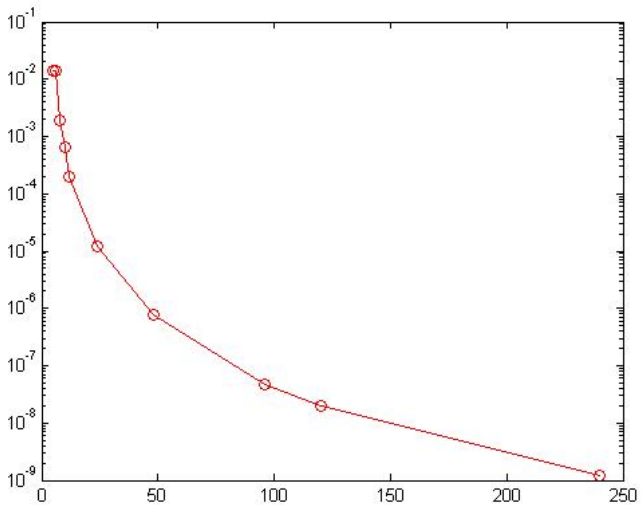


Fig.3.19. Variaţia erorii maxime de interpolare în funcţie de f_e/ f_s.

3.8. Concluzii

Funcţiile spline sunt un instrument eficient în realizarea unor aplicaţii de interpolare de bună calitate. Acest lucru se datorează proprietăţilor funcţiilor spline, care sunt polinoame de gradul *n* pe subintervale adiacente, care se unesc în noduri, funcţia şi şi primele *n*-1 derivate ale sale fiind continue.

În ultimele 2 decenii, funcţiile spline au fost utilizate tot mai des în aplicaţii de prelucrare numerică a imaginilor. Au fost elaboraţi numeroşi algoritmi de interpolare ce folosesc mai multe tipuri de funcţii spline (B-spline, spline polinomiale, spline cardinale) sau combinaţii ale acestora. Toţi algoritmii au fost realizaţi utilizînd conceptul de interpolare generalizată şi se bazează pe algoritmul care foloseşte transformata B-spline directă pentru calcularea coeficienţilor şi transformata B-spline inversă pentru determinarea eşantioanelor semnalului interpolat. Cei mai des întâlniţi sunt algoritmii care folosesc funcţii de gradul 3.

Aceşti algoritmi sunt integraţi în aplicaţii specifice prelucrării imaginilor, precum redimensionarea sau aplicarea unor transformări geometrice, şi conduc la obţinerea unor rezultate foarte bune cu costuri de implementare mici.

Algoritmul rapid de interpolare spline cubică (algoritmul Unser) a fost aplicat mai multor seturi de eşantioane provenite de la semnale periodice (sinus, cosinus, triunghi) sau neperiodice (treaptă), semnale continue sau cu puncte de discontinuitate. Pentru unele dintre ele s-au folosit mai multe şiruri corespunzătoare unor situaţii în care diferă frecvenţa de eşantionare (de exemplu, M=12 şi M=120 pentru sinus şi respectiv, M=8 şi M=16, pentru triunghi). Din analizarea coeficienţilor şi a erorilor de interpolare s-au desprins câteva concluzii, sintetizate în continuare.

Valorile coeficienţilor depind de frecvenţa cu care este eşantionat semnalul de intrare.

Valorile coeficienţilor urmăresc variaţia semnalului şi sunt apropiate de valorile eşantioanelor acestuia. Cu cât se creşte frecvenţa de eşantionare, cu atât coeficienţii sunt mai apropiaţi de valorile semnalului.

Coeficienţii obţinuţi pentru una sau mai multe perioade ale unui semnal sunt aceeaşi cu excepţia primilor şi ultimilor coeficienţi din şir. Excepţiile se datorează condiţiilor iniţiale impuse la calcularea coeficienţilor. Acestea duc la apariţia aşa numitelor „erori la capete”. Apariţia acestor erori nu este precizată nicăierei în lucrările studiate. Însă, în aceste lucrări, exemplele sunt prezentate pentru prelucrarea unor imagini. În general, imaginile sunt reprezentate printr-un volum mare de date, ceea ce poate duce la ignorarea erorilor de capete, ele fiind nesemnificative în raport cu multitudinea de eşantioane prelucrate.

Se poate spune că în noduri, interpolarea este exactă deoarece erorile de interpolare în aceste puncte au valori foarte mici. Mărimea erorilor de interpolare din punctele dintre noduri depinde de frecvenţa de eşantionare a semnalului de intrare (cât de dese sunt nodurile). Cu cât frecvenţa de eşantionare este mai mare, cu atât mai mici sunt aceste erori.

4. NOI ALGORITMI DE DETERMINARE A COEFICIENŢILOR B-SPLINE

4.1. Introducere

În capitolul anterior au fost prezentate mai multe metode de interpolare cu funcţii spline care au la bază algoritmii dezvoltaţi de M. Unser şi colaboratorii. Prelucrarea semnalelor se face în contextul de interpolare generalizată care presupune parcurgerea a 2 paşi: prefiltrare şi interpolare (sau sinteză). Majoritatea studiilor analizează algoritmii Unser prin diverse exemple şi propun noi metode pentru implementarea celui de-al doilea pas, cel de sinteză. Însă, partea de prefiltrare este realizată prin metoda propusă de Unser şi colaboratorii. Astfel, din eşantioanele de intrare, coeficienţii B-spline sunt obţinuţi la ieşirea filtrului B-spline indirect (filtru IIR). Mai apoi, aceşti coeficienţi sunt aduşi la intrarea unor noi sisteme propuse pentru interpolare sau aproximare.

Algoritmul nu poate fi utilizat pentru prelucrarea semnalelor în timp real. În una dintre lucrările amintite [123] se aduce în discuţie acest fapt şi se propun variante de modificare ale filtrului B-spline invers astfel încât să se elimine acest inconvenient. Filtrul IIR este înlocuit cu filtre cu răspuns finit la impuls (FIR) obţinute prin trunchierea răspunsului în frecvenţă al filtrului iniţial.

Inconvenientul utilizării filtrului B-spline indirect este dat de faptul că presupune utilizarea încă de la început a întregului şir de eşantioane din semnalul de intrare. Coeficienţii sunt calculaţi printr-un algoritm compus din 2 procese recursive: unul cauzal şi unul anticauzal.

Procesul de determinare a coeficienţilor B-spline presupune şi stabilirea unor condiţii iniţiale pe baza prelungirii semnalului de intrare. Pentru funcţii care nu sunt periodice, sau pentru funcţii impare, acest tip de oglindire duce la introducerea unor erori suplimentare.

În continuare se caută noi metode de calculare a coeficienţilor B-spline şi de determinare a condiţiilor iniţiale care să nu necesite prelungirea semnalului şi să poată fi utilizate pentru prelucrarea numerică a semnalelor în timp real. Se porneşte de la algoritmul rapid de interpolare cu funcţii B-spline cubice prezentat în subcapitolul 3.7. Din procesul descris de algoritmul Unser se păstrează partea a doua, cea de interpolare. Pentru sistemul ce realizează prefiltrarea, se vor căuta noi funcţii de transfer.

În figurile 4.1 şi 4.2 sunt redate procesul de reconstrucţie, respectiv interpolare cu un factor m ale unui semnal $\{y(k)\}$. Pasul de prefiltrare va fi implementat printr-un filtru nou, cu funcţia de transfer $H_{pf}(z)$, iar partea de reconstrucţie şi interpolare sa face cu filtre B-spline indirecte.

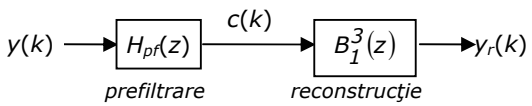


Fig. 4.1. Reconstrucţia semnalului.

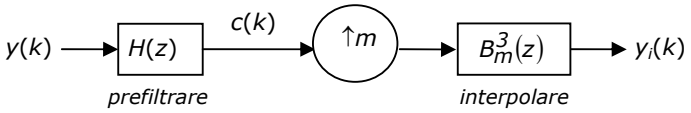


Fig. 4.2. Interpolarea semnalului cu un factor m=2.

Metoda clasică de interpolare cu funcţii spline cubice presupune utilizarea şi a valorilor derivatelor funcţiei în noduri [60, 65]. În studii recente din prelucrarea semnalelor şi a imaginilor se utilizează diferenţele finite [35, 62], diferenţele divizate [63] şi integrarea numerică [49] a funcţiilor B-spline pentru realizarea unor filtre folosite în prelucrarea semnalelor. Pornind de la aceste observaţii, se vor analiza relaţiile dintre coeficienţi, valorile funcţiei şi ale derivatelor sale de ordinul 1 şi 2 în noduri.

Pe baza acestor relaţii se dezvoltă mai mulţi algoritmi de interpolare. Fiecare algoritm în parte va fi implementat în *Matlab* şi testat pe o serie de semnale reprezentative, bine cunoscute. Dintre acestea amintim: semnal sinusoidal, cosinus, dreaptă (semnale continue), triunghiular, semnal treaptă (semnale ce prezintă discontinuităţi). Câteva rezultate considerate semnificative vor fi prezentate, comparate şi discutate.

Deoarece se lucrează cu semnale discrete, trebuie găsite metode de derivare numerică. Vor fi luate în considerare 3 cazuri care ne vor permite să determinăm valorile derivatelor funcţiei în noduri.

4.2. Derivare numerică

În literatura de specialitate sunt întâlnite mai multe modalităţi de definire şi evaluare a procedeului de derivare numerică. Vor fi prezentate şi utilizate 3 dintre acestea:

1. metoda clasică de calcul a diferenţelor finite;
2. metoda de derivare numerică descrisă de O. Stănăşilă;
3. evaluarea diferenţei finite centrale a unui polinom care trece prin 5 puncte.

4.2.1. Diferenţe finite

Fie $f(u)$ o funcţie reală definită pe o mulţime de puncte echidistante (u, $u+h$, $u+2h$, $u+3h$, ..., $u+jh$), unde $h>0$. Se defineşte diferenţa finită de ordinul 1 a funcţiei $f(u)$ în punctul u [27, 45, 93]:

$$\Delta f(u) = \Delta f_u = f(u+h) - f(u) \tag{4.1}$$

Diferenţa finită de ordinul al doilea este:

$$\Delta^2 f(u) = \Delta f(u+h) - \Delta f(u) = f(u+2h) - 2f(u+h) + f(u) \tag{4.2}$$

Acestea se mai numesc şi diferenţe finite la dreapta, sau progresive. Se pot defini şi diferenţe finite la stânga (regresive) [45]:

$$\nabla f(u) = f(u) - f(u-h) \tag{4.3}$$

$$\nabla^2 f(u) = f(u) - 2f(u-h) + f(u-2h)$$

Gh. Micula defineşte diferenţele divizate pentru o funcţie $f(u)$, definită pe o mulţime de puncte distincte (u_0, u_1, ..., u_N) [60]:

$$(Df)(u_r) = \frac{f(u_{r+1}) - f(u_r)}{u_{r+1} - u_r}, 0 < r < N \tag{4.4}$$

Diferenţa divizată de ordinul k:

$$\left(D^k f\right)(u_r) = \frac{\left(D^{k-1} f\right)(u_{r+1}) - \left(D^{k-1} f\right)(u_r)}{u_{r+k} - u_r} \tag{4.5}$$

În cazul în care punctele u_r sunt echidistante este vorba de diferenţe finite şi definiţiile sunt echivalente cu (4.1) şi (4.2).

În continuare vor fi utilizate diferenţele finite, deoarece se lucrează doar cu eşantioane prelevate în puncte echidistante şi de spaţiere unitară. Se vor folosi definiţiile pentru diferenţe finite la dreapta (progresive).

4.2.2. Derivare numerică prin metoda Stănăşilă

Pentru procesul de derivare numerică, O. Stănăşilă propune o formulare puţin diferită [87]. Fie $f \in C^2[a,b]$ şi u, h alese astfel încât $a < u - h < u < u + h < b$. Pentru o astfel de funcţie continuă pe un interval şi cu derivatele continue, se pot aproxima:

- derivata la stânga $f'(u) \cong \frac{f(u) - f(u-h)}{h}$

- derivata la dreapta $f'(u) \cong \frac{f(u+h) - f(u)}{h}$

Atunci, derivata centrată de ordinul 1 este:

$$f'(u) \cong \frac{f(u+h) - f(u-h)}{2h} \tag{4.6}$$

şi derivata centrată de ordinul 2:

$$f''(u) \cong \frac{f(u+h) - 2f(u) + f(u-h)}{h^2} \tag{4.7}$$

Cum am mai spus, funcţiile considerate au noduri echidistante, cu spaţiere unitară. Astfel, relaţiile de mai sus devin:

$$f'(u) \cong \frac{f(u+1)-f(u-1)}{2} \tag{4.8}$$

$$f''(u) \cong f(u+1)-2f(u)+f(u-1) \tag{4.9}$$

Derivatele astfel calculate sunt de fapt diferenţele finite centrale ale unui polinom ce trece prin 3 puncte.

Se consideră următorul polinom:

$$f(u)=f(u_{k-1})+a(u-u_{k-1})+b(u-u_{k-1})^2 \tag{4.10}$$

Acesta trece prin 3 puncte u_{k-1}, u_k, u_{k+1}, puncte pe care le considerăm egal distanţate, cu spaţiere unitară: $u_{k+1}-u_k=1$.

Pentru $u=u_{k-1}$ se notează $f(u_{k-1})$ cu $f(k-1)$.

$$u=u_k \Rightarrow f(k)=f(k-1)+a+b \tag{4.11}$$

$$u=u_{k+1} \Rightarrow f(k+1)=f(k-1)+2a+4b \tag{4.12}$$

Din (4.11) şi (4.12) se obţine:

$$4f(k)-f(k+1)=3f(k-1)+2a$$

$$\Rightarrow \quad a=(4f(k)-f(k+1)-3f(k-1))/2$$

$$\Rightarrow$$

$$\Rightarrow \quad b=f(k)-f(k-1)-a \;\Rightarrow b=(f(k+1)-2f(k)+f(k-1))/2$$

Se derivează (4.10) şi se obţine:

$$f'(u)=a+2b(u-u_{k-1})$$

$$\Rightarrow \quad f'(k)=a+2b$$

$$\Rightarrow \quad f'(k)=(4f(k)-f(k+1)-3f(k-1)+2f(k+1)-4f(k)+2f(k-1))/2$$

$$\Rightarrow \quad f'(k)=[f(k+1)-f(k-1)]/2$$

Se determină derivata de ordinul 2:

$$f''(k)=2b$$

$$\Rightarrow \quad f''(k)=f(k+1)-2f(k)+f(k-1)$$

Relaţiile obţinute pentru polinomul ce trece prin 3 puncte sunt chiar cele date de formulele (4.8) şi (4.9).

4.2.3. Diferenţa finită centrală a unui polinom ce trece prin 5 puncte

Micula a demonstrat că pentru condiţii mai tari de continuitate se îmbunătăţesc proprietăţile de convergenţă ale funcţiei spline de interpolare [60]. Se consideră $f \in C^4[a,b]$ o funcţie polinomială continuă şi cu derivatele până la ordinul 4 continue. Funcţia poate fi descrisă prin polinomul:

$$f(u) = a + bu + du^2 + eu^3 + gu^4 \tag{4.13}$$

Această funcţie este polinomială pe porţiuni, aşa că poate fi analizată pe intervale scurte. Se evaluează valorile funcţiei şi ale derivatelor sale de ordinul 1 şi 2 în intervalul [0; 4].

Parametrii polinomului sunt:

$$\begin{cases} a = f(0) \\ b = \left(-25f(0) + 48f(1) - 36f(2) + 16f(3) - 3f(4)\right)/12 \\ d = \left(35f(0) - 104f(1) + 114f(2) - 56f(3) + 11f(4)\right)/24 \\ e = \left(-5f(0) + 18f(1) - 24f(2) + 14f(3) - 3f(4)\right)/12 \\ g = \left(f(0) - 4f(1) + 6f(2) - 4f(3) + f(4)\right)/24 \end{cases}$$

Derivata de ordinul 1 a polinomului este:

$$f'(u) = b + 2du + 3eu^2 + 4gu^3$$

$$\Rightarrow \quad f'(1) = \frac{-3f(0) - 10f(1) + 18f(2) + f(4)}{12} \tag{4.14}$$

Se calculează şi derivata a doua:

$$f''(u) = 2d + 6eu + 12gu^2$$

$$\Rightarrow \quad f''(2) = \frac{-\left(f(0) + f(4)\right) + 16\left(f(1) + f(3)\right) - 30f(2)}{12} \tag{4.15}$$

Determinarea valorilor pentru $f'(1)$ şi $f''(2)$ interesează pentru calcularea valorilor iniţiale în subcapitolul 4.4 [90, 91].

În mod analog se poate evalua polinomul pe un interval oarecare de 5 puncte centrat în punctul k: [k-2, k-1, k, k+1, k+2]. Pentru diferenţele finite centrale de ordinul 1 şi 2 se obţin formulele [53, 90]:

$$f'(k) = \frac{f(k-2) - 8f(k-1) + 8f(k+1) - f(k+2)}{12} \tag{4.16}$$

$$f''(k) = \frac{-\left(f(k-2) + f(k+2)\right) + 16\left(f(k-1) + f(k+1)\right) - 30f(k)}{12} \tag{4.17}$$

Au fost prezentate 3 metode ce permit calcularea valorilor derivatelor de ordinul 1 şi 2 ale funcţiei în raport cu valorile funcţiei în noduri. Deoarece funcţia $f(u)$ ia valorile semnalului de intrare în noduri, derivatele se pot calcula astfel doar pe baza eşantioanelor din şirul de intrare $\{y(k)\}$.

4.3. Relaţiile dintre coeficienţi, funcţie şi derivatele sale

Se porneşte de la presupunerea că semnalul de intrare este aproximat printr-o funcţie spline cubică $f(x)$. Aceasta este o funcţie polinomială de gradul 3 pe porţiuni. Atât funcţia, cât şi derivatele sale de ordinul 1 şi 2, sunt continue în noduri. Nodurile acestei funcţii sunt reprezentate chiar de punctele k în care sunt date valorile eşantioanelor de intrare $\{y(k)\}$, cu k=0,1, ... , N-1.

Funcţia poate fi caracterizată prin coeficienţii B-spline astfel:

$$f(x) = \sum_{l \in Z} c(l)\beta^3(x-l)\Big|_{x=k} = y(k) \tag{4.18}$$

Dat fiind faptul că se lucrează cu semnale în timp discret, se va utiliza funcţia B-spline discretă $b^3(k)$ dată de relaţia (3.70). Pentru funcţii spline cubice, coeficienţii sunt obţinuţi prin prefiltrare, operaţie prezentată în figura (3.6) şi descrisă de ecuaţia (3.45). În continuare se caută determinarea relaţiilor între coeficienţi, valorile funcţiei şi ale derivatelor sale de ordinul 1 şi 2. Noul şir de coeficienţi se va nota cu $c_s(k)$.

În fiecare nod k se poate scrie relaţia între funcţia spline cubică şi coeficienţii săi B-spline [91]:

$$6f(k) = 4c_s(k) + c_s(k-1) + c_s(k+1) \tag{4.19}$$

Datorită proprietăţilor funcţiilor B-spline la derivare (3.30), pentru derivatele de ordinul întâi şi doi se pot scrie relaţiile:

$$f'(k) = \left(b^{3\prime}\right) * c_s(k) \tag{4.20}$$

$$f''(k) = \left(b^{3\prime\prime}\right) * c_s(k) \tag{4.21}$$

Valorile derivatelor funcţiei B-spline discrete se determină din relaţiile (3.34) şi (3.35). Relaţiile între derivatele funcţiei $f(k)$ şi coeficienţi sunt:

$$f'(k) = 0c_s(k) - \frac{1}{2}c_s(k-1) + \frac{1}{2}c_s(k+1)$$

$$\Rightarrow \quad 2f'(k) = c_s(k+1) - c_s(k-1) \tag{4.22}$$

$$\Rightarrow \quad c_s(k+1) = 2f'(k) + c_s(k-1) \tag{4.23}$$

Pentru a stabili o nouă procedură de calcul a coeficienţilor pe baza relaţiei (4.23) este nevoie şi de precizarea unor condiţii iniţiale. Înainte de a vedea cum se pot determina primele valori ale şirului de coeficienţi, se evaluează şi derivata de ordinul 2 a funcţiei B-spline cubică. Se poate scrie:

$$f''(k) = -2c_s(k) + c_s(k-1) + c_s(k+1) \tag{4.24}$$

Din relaţiile (4.24) şi (4.19) se obţine:

$$c_s(k) = f(k) - \frac{f''(k)}{6} \tag{4.25}$$

Pentru stabilirea primelor valori ale coeficienţilor, iniţializarea şirului nu se poate face în punctul k=0 deoarece nu avem informaţii privind valorile funcţiei de intrare înainte de acest punct sau caracteristici ale acestei funcţii. Se evaluează relaţiile (4.22), (4.24) şi (4.19) în punctul k=2:

$$2f'(2) = c_s(3) - c_s(1)$$

$$f''(2) = -2c_s(2) + c_s(1) + c_s(3) \Rightarrow c_s(1) + c_s(3) = f''(2) + 2c_s(2)$$

$$6f(2) = 4c_s(2) + c_s(1) + c_s(3) = 4c_s(2) + f''(2) + 2c_s(2) = 6c_s(2) + f''(2)$$

$$\Rightarrow \quad c_s(2) = f(2) - \frac{f''(2)}{6} \tag{4.26}$$

În mod similar, pentru k=1 se poate scrie:

$$2f'(1) = c_s(2) - c_s(0)$$

$$\Rightarrow \quad c_s(0) = c_s(2) - 2f'(1) \tag{4.27}$$

$$6f(1) = 4c_s(1) + c_s(0) + c_s(2)$$

$$\Rightarrow \quad c_s(1) = \frac{6f(1) - c_s(0) - c_s(2)}{4} \tag{4.28}$$

Am arătat că se pot calcula primii trei coeficienţi făcând iniţializarea în punctul 2. Coeficienţii $c_s(2)$, $c_s(0)$ şi $c_s(1)$, se calculează pe baza relaţiilor (4.26), (4.27) şi (4.28) utilizând valorile funcţiei şi ale derivatelor sale în punctele respective [91]. Valorile derivatelor în aceste puncte se vor determina cu una dintre metodele descrise în subcapitolul 4.2.

Pentru exemplificare s-au determinat primii 3 coeficienţi pentru un semnal $y(k)=sin(2\pi k/M)$ unde M=12, apoi M=120. În formulele (4.26) şi (4.27) s-au utilizat derivatele unui polinom ce trece prin 5 puncte. Coeficienţii au fost comparaţi cu valorile obţinute la ieşirea filtrului B-spline direct propus de M. Unser în 2 cazuri: k_0=7 şi $k_0 \to \infty$ [91]. Pentru $k_0 \to \infty$ s-a calculat $c^+(0)$ din formula (3.64) pentru

semnalul obţinut prin oglindire la ambele capete cu un număr de termeni egal cu o perioadă din semnal. Acest tip de oglindire duce la apariţia de discontinuităţi pentru semnalele impare, aşa cum este sinus. Rezultatele sunt prezentate în tabelul 4.1.

Tabelul 4.1. Valorile primilor 3 coeficienţi pentru $y(k)=sin(2\pi k/M)$

M	k	$c(k)$		$c_s(k)$
12	0	$k_0=7$	$-3{,}0213\cdot10^{-1}$	$-3{,}5163\cdot10^{-3}$
		$k_0\to\infty$	0	
	1	$k_0=7$	$6{,}0433\cdot10^{-1}$	$5{,}2448\cdot10^{-1}$
		$k_0\to\infty$	$5{,}2337\cdot10^{-1}$	
	2	$k_0=7$	$8{,}8481\cdot10^{-1}$	$9{,}0556\cdot10^{-1}$
		$k_0\to\infty$	$9{,}0650\cdot10^{-1}$	
120	0	$k_0=7$	$-3{,}0244\cdot10^{-2}$	0
		$k_0\to\infty$	0	
	1	$k_0=7$	$6{,}0463\cdot10^{-2}$	$5{,}2359\cdot10^{-2}$
		$k_0\to\infty$	$5{,}2359\cdot10^{-2}$	
	2	$k_0=7$	$1{,}0240\cdot10^{-1}$	$1{,}0457\cdot10^{-1}$
		$k_0\to\infty$	$1{,}0457\cdot10^{-1}$	

După cum se poate vedea, noile valori sunt mult mai apropiate de cele calculate pentru cazul ideal în care $k_0\to\infty$. Cu noua modalitate de determinare a coeficienţilor iniţiali se elimină inconvenientele datorate prelungirii semnalului prin oglindire.

Anterior s-a descris o modalitate de determinare a condiţiilor iniţiale pentru calcularea coeficienţilor. În continuare se vor prezenta mai multe metode de calcul a şirului de coeficienţi, utilizând aceste prime valori.

4.4. Algoritmul zero (neconvergent)

După determinarea valorilor iniţiale ale coeficienţilor $c_s(0)$, $c_s(1)$ şi $c_s(2)$ trebuie stabilită modalitatea de calcul pentru ceilalţi coeficienţi. Pornim de la reprezentarea filtrului B-spline direct (3.57). M. Unser propune implementarea acestui filtru prin cascadarea a 2 filtre [100]: unul cauzal şi cel de-al doilea anticauzal.

Autoarea acestei teze propune următoarea abordare: nu se descompune sistemul în 2 filtre, ci este reprezentat printr-unul singur, a cărui funcţie de transfer este:

$$H_0(z)=\frac{6}{z+4+z^{-1}}=\frac{6z}{z^2+4z+1} \tag{4.29}$$

Coeficienţii determinaţi cu noul algoritm vor fi notaţi cu $c_0(k)$. În acest caz se poate scrie:

$$6y(k)=4\,c_0(k)+c_0(k-1)+c_0(k+1) \tag{4.30}$$

De aici se determină:

$$c_0(k+1) = 6y(k) - 4\,c_0(k) - c_0(k-1) \tag{4.31}$$

Pentru orice semnal dat prin eşantioanele $\{y(k)\}$, cu k=0, 1,…, N-1, algoritmul de interpolare, denumit în continuare *algoritmul zero*, presupune parcurgerea următorilor paşi:

- se determină valorile $c_0(2)$, $c_0(0)$ şi $c_0(1)$ din formulele (4.26), (4.27) şi (4.28), în care derivatele sunt aproximate printr-una din metodele prezentate în subcapitolul 4.3;
- pentru k=2, 3,…, N-1, se calculează coeficienţii $c_0(k)$ cu relaţia (4.31);
- pentru reconstrucţia semnalului $y_r(k)$ se aplică filtrul B-spline indirect $B_1^3(z)$ (conform schemei din figura 4.1);
- pentru interpolarea semnalului cu un factor m se realizează mai întâi supraeşantionarea şirului de coeficienţi $c_{0m}(i)$ conform relaţiei (3.48), unde i=0, 1, ...,(Nm-1);
- semnalul interpolat $y_i(k)$ este obţinut la ieşirea filtrului B-spline indirect $B_m^3(z)$.

Procesul de interpolare este redat schematic în figura 4.2, unde operaţia de prefiltrare este realizată cu ajutorul filtrului care are funcţia de transfer $H_0(z)$ (4.29).

De exemplu, având valorile pentru primii 3 coeficienţi: $c_0(0)$, $c_0(1)$ şi $c_0(2)$, putem calcula în continuare elementele şirului $c_0(k)$ [89]:

$$c_0(3) = 6y(2) - 4c_0(2) - c_0(1)$$

$$c_0(4) = 6y(3) - 4c_0(3) - c_0(2) = 6y(3) - 4(6y(2) - 4c_0(2) - c_0(1)) - c_0(2) = \\ = 6y(3) - 24y(2) + 15c_0(2) + 4c_0(1)$$

$$c_0(5) = 6y(4) - 4c_0(4) - c_0(3) = 6y(4) - 4(6y(3) - 24y(2) + 15c_0(2) + \\ +4c_0(1)) - 6y(2) - 4c_0(2) - c_0(1) = 6y(4) - 24y(3) + 90y(2) - \\ -56c_0(2) - 15c_0(1)$$

Teoretic, dacă se continuă în acest mod, se observă că orice coeficient poate fi calculat pe baza eşantioanelor semnalului de intrare şi a coeficienţilor $c_0(1)$ şi $c_0(2)$ [89]. Practic, fiecare coeficient $c_0(k)$ cu k>2 se determină pe baza celor 2 coeficienţi anteriori $c_0(k-1)$ şi $c_0(k-2)$, şi a eşantionului anterior de la intrare $y(k-1)$. Astfel, erorile de determinare a fiecărui coeficient se propagă în şir, influenţând valorile următoare.

Presupunem că determinarea coeficienţilor $c_0(1)$ şi $c_0(2)$ se face cu erorile absolute $e(1)$ şi $e(2)$. La adunare sau scădere, eroarea absolută maximă a rezultatului e dată de suma erorilor absolute maxime ale termenilor [94]. La determinarea coeficientului următor $c_0(3)$ se poate scrie:

$$c_0(3) \pm e(3) = 6y(2) - 4[c_0(2) \pm e(2)] - [c_0(1) \pm e(1)]$$

$$\Rightarrow \qquad e(3) \approx \pm[4e(2) + e(1)]$$

Calculând mai departe, se obţine:

$$e(4) \approx \pm[4e(3) + e(2)]$$

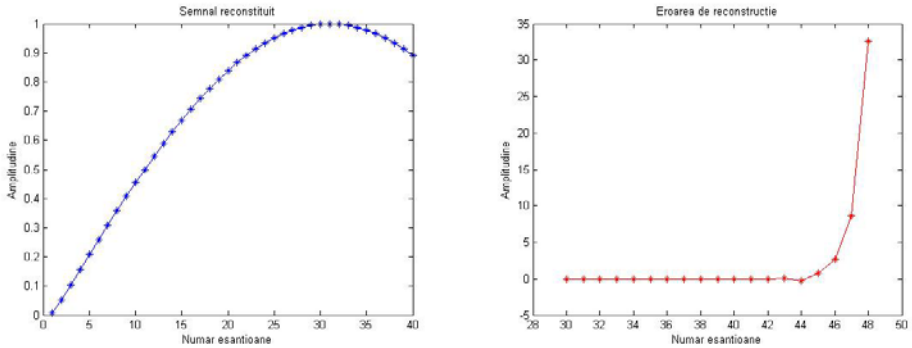
Se poate spune că pentru orice coeficient $c_s(k)$ avem:

$$e(k) \approx \pm[4e(k-1) + e(k-2)] \tag{4.32}$$

După cum se poate observa, la determinarea unui nou coeficient, eroarea absolută poate să crească de aproximativ 4 ori faţă de eroarea coeficientului anterior. Astfel, după un anumit număr de iteraţii, valorile coeficienţilor pot să crească foarte mult, fiind însoţite de erori considerabile.

Algoritmul a fost implementat în *Matlab* şi testat pe mai multe tipuri semnale (sinus, cosinus, dreaptă). Pentru un şir de eşantioane de intrare $\{y(k)\}$, cu k=0, 1,…, N-1, s-a determinat şirul coeficienţilor $\{c_s(k)\}$, apoi s-a reconstruit semnalul $\{y_r(k)\}$. Semnalul reconstruit a fost comparat cu cel de la intrare, fiind calculată eroarea de reconstrucţie. Programul, împreună cu câteva rezultate, se găsesc în *Anexa 2*.

Ca exemplu se iau $y(k)=sin(2\pi k/M)$, unde M=120, N=361, şi k=0, 1,…, N-1. S-a observat că pentru primele 20-30 puncte coeficienţii calculaţi cu noua metodă au valori apropiate de cele ale coeficienţilor calculaţi cu algoritmul lui M. Unser. După aceea, valorile cresc foarte mult, ducând la creşterea erorii de reconstrucţie a semnalului. În figura 4.3 sunt prezentate partea de început a semnalului reconstruit (4.3.a)) şi o secvenţă din eroarea de reconstrucţie (4.3.b)).



a) semnal reconstruit b) eroarea de reconstrucţie

Fig. 4.3. Rezultate pentru reconstrucţia semnalului sinusoidal cu *algoritmul zero*.

Rezultate similare s-au obţinut şi pentru semnalul $y(k)=cos(2\pi k/M)$, cu aceleaşi valori pentru M=120 şi N=361. Apoi s-au folosit aceleaşi semnale, dar cu eşantioane corespunzătoare unei perioade de eşantionare de 10 ori mai mare (M=12). Erorile sunt neglijabile pentru primele aproximativ 10 eşantioane, apoi cresc considerabil cu fiecare iteraţie [89].

Studiind rezultatele, s-au formulat următoarele observaţii:

- la începutul şirului, pentru un anumit număr de puncte, valorile coeficienţilor sunt apropiate de cele calculate cu algoritmul Unser;

- erorile cresc foarte mult apoi devin din ce în ce mai mari cu fiecare iteraţie;
- erorile cresc mai repede dacă perioada de eşantionare a semnalului este mai mare.

Relaţia de calcul a coeficienţilor poate fi folosită sub această formă dacă, după un anumit număr de iteraţii, se face reiniţializarea şirului.

Algoritmul a fost aplicat şi unui semnal care nu prezintă variaţii, semnal constant (eşantioane corespunzătoare unei drepte). Rezultatele nu sunt afectate major de propagarea erorilor, aşa cum se întâmplă pentru semnale care prezintă variaţii (precum sinus şi cosinus).

Algoritmul, implementat în această formă, este divergent. Acest lucru poate fi demonstrat şi din analiza proprietăţilor sistemului folosit pentru prefiltrare.

Şirul eşantioanelor $\{y(k)\}$ este adus la intrarea unui filtru numeric cu funcţia de transfer $H_0(z)$ dată de (4.29), obţinând la ieşire şirul coeficienţilor. $H_0(z)$ este un filtru cu răspuns infinit la impuls, ce poate fi scris şi sub forma:

$$H_0(z) = \frac{6z}{z^2 + 4z + 1} = \frac{6(z - z_0)}{(z - z_{p1})(z - z_{p2})} \tag{4.33}$$

Elementele z_{p1} şi z_{p2} reprezintă polii sistemului. Valorile acestora trebuie să fie în interiorul cercului unitate pentru ca sistemul cauzal să fie stabil [64, 86]:

$$|z_{pi}| < 1 \qquad \text{pentru } i=1;2.$$

În cazul analizat: $z_{p1}\approx$-0,27 şi $z_{p2}\approx$-3,73, deci sistemul este instabil. Pentru rezolvarea acestei probleme, M. Unser realizează descompunerea sistemului în 2 filtre: unul cauzal şi unul anticauzal [100, 108, 112].

Concluzii:

- Operaţiile de interpolare se fac de obicei pe seturi de date de lungime mare. Algoritmul zero nu poate fi aplicat în aceste cazuri.
- Tehnica poate fi utilizată cu succes pentru şiruri scurte, de maximum 10-15 eşantioane, în funcţie şi de frecvenţa de eşantionare.
- Se poate folosi ca metodă adiţională la alţi algoritmi pentru a îmbunătăţi rezultatele.

4.5. Algoritmi care utilizează derivata 1

4.5.1. Consideraţii generale

Metoda clasică de interpolare cu funcţii spline utilizează şi derivatele semnalului în punctele de interpolare [60]. Se va folosi derivarea numerică şi în stabilirea unor noi algoritmi pentru determinarea coeficienţilor B-spline. Interpolarea devine de tip Hermite prin faptul că se impun de la început valori ale derivatelor funcţiei în noduri [60, 63].

Plecând de la proprietăţile funcţiei B-spline cubică şi ale derivatelor sale, în subcapitolul 4.3 s-a determinat o relaţie (4.23) între coeficienţi şi valorile derivatei de ordinul 1. În orice punct k se poate determina valoarea coeficientului pe baza

coeficientului calculat în punctul *k*-2 şi a valorii derivatei de ordinul 1 aproximate în punctul *k*-1:

$$c_S(k) = 2f'(k-1) + c_S(k-2) \tag{4.34}$$

Pentru iniţializarea şirului de coeficienţi, valorile $c_S(0)$, $c_S(1)$ şi $c_S(2)$ vor fi calculate pe baza formulelor (4.26), (4.27) şi (4.28) din subcapitolul 4.2.

Mai departe se vor prezenta 3 algoritmi originali în care, pentru determinarea valorilor derivatelor în noduri, se utilizează fiecare din metodele de derivare numerică prezentate în subcapitolul 4.2.

4.5.2. Algoritmul 1A

Pentru determinarea valorilor *f'(u)* se utilizează definiţia clasică a diferenţei finite [45, 93], prezentată în subcapitolul 4.2.1. Funcţia *f*(*u*) aproximează semnalul de intrare şi în orice punct *k*: *f*(*k*)=*y*(*k*). Coeficienţii determinaţi în acest caz vor fi notaţi cu $c_{1A}(k)$. În aceste condiţii, relaţia (4.34) devine:

$$c_{1A}(k) = 2[y(k) - y(k-1)] + c_{1A}(k-2) \tag{4.35}$$

Aceasta mai poate fi scrisă şi sub forma:

$$c_{1A}(k+1) - c_{1A}(k-1) = 2[y(k+1) - y(k)]$$

$$\Rightarrow \qquad H_{1A}(z) = \frac{2(z-1)}{z - z^{-1}} \tag{4.36}$$

Condiţiile iniţiale se calculează utilizând aceleaşi formule pentru diferenţele finite:

$$c_{1A}(2) = f(2) - \frac{f''(2)}{6} = y(2) - \frac{y(4) - 2y(3) + y(2)}{6} \tag{4.37}$$

$$c_{1A}(0) = c_{1A}(2) - 2f'(1) = c_{1A}(2) - 2[y(2) - y(1)] \tag{4.38}$$

$$c_{1A}(1) = \frac{6y(1) - c_{1A}(0) - c_{1A}(2)}{4} \tag{4.39}$$

Algoritmul 1A presupune parcurgerea următorilor paşi:

- din eşantioanele semnalului de intrare {*y*(*k*)}, cu *k*=0, 1,…, *N*-1, se determină $c_{1A}(2)$, $c_{1A}(0)$ şi $c_{1A}(1)$ cu formulele (4.37), (4.38) şi (4.39);
- pentru *k*>2 se calculează fiecare coeficient $c_{1A}(k)$ cu relaţia (4.35);
- pentru reconstrucţia semnalului {$y_r(k)$}, şirului de coeficienţi {$c_{1A}(k)$} i se aplică filtrul B-spline indirect $B_1^3(z)$;

- pentru interpolarea semnalului $\{y_i(k)\}$ cu un factor m se face supraeşantionarea şirului de coeficienţi $c_{1Am}(i)$ (3.48), care apoi se aduce la intrarea filtrului B-spline indirect $B_m^3(z)$.

Pentru partea de prefiltrare, sistemul are funcţia de transfer dată de (4.36) şi reprezintă un filtru cu răspuns infinit la impuls (IIR). Polii sistemului se regăsesc pe cercul unitate, fapt ce poate duce la apariţia de oscilaţii [64, 86].

4.5.3. Algoritmul 1D

Încă de la început s-a considerat că eşantioanele $\{y(k)\}$ pentru care se face interpolarea au fost obţinute prin eşantionarea uniformă a unui semnal. Astfel, datele de intrare sunt echidistante şi cu spaţiere unitară. În acest caz, pentru derivarea numerică se pot utiliza şi definiţiile (4.6) şi (4.7) propuse de O. Stănăşilă [87], atât pentru formula recursivă de calcul a coeficienţilor, cât şi pentru determinarea valorilor iniţiale ale şirului. Coeficienţii determinaţi în acest mod vor fi notaţi cu $c_{1D}(k)$.

Algoritmul 1D permite calcularea coeficienţilor $\{c_{1D}(k)\}$ şi a semnalelor reconstruit $\{y_r(k)\}$, respectiv interpolat $\{y_i(k)\}$ prin parcurgerea următorilor paşi:

- determinarea valorilor primilor 3 coeficienţi:

$$c_{1D}(2) = f(2) - \frac{f''(2)}{6} = y(2) - \frac{y(3) - 2y(2) + y(1)}{6} \tag{4.40}$$

$$c_{1D}(0) = c_{1D}(2) - 2f'(1) = c_{1D}(2) - y(2) + y(0) \tag{4.41}$$

$$c_{1D}(1) = \frac{6y(1) - c_{1D}(0) - c_{1D}(2)}{4} \tag{4.42}$$

- fiecare coeficient $c_{1D}(k)$, pentru $k>2$ se calculează cu relaţia:

$$c_{1D}(k) = y(k) - y(k-2) + c_{1D}(k-2) \tag{4.43}$$

- pentru reconstrucţia semnalului, şirului de coeficienţi $\{c_{1D}(k)\}$ i se aplică filtrul B-spline indirect $B_1^3(z)$;
- pentru interpolarea semnalului cu un factor m se face supraeşantionarea şirului de coeficienţi $c_{1Dm}(i)$, care apoi se aduce la intrarea filtrului B-spline indirect $B_m^3(z)$.

Relaţia (4.43) mai poate fi scrisă şi sub forma:

$$c_{1D}(k+1) - c_{1D}(k-1) = y(k+1) - y(k-1) \tag{4.44}$$

Pe baza relaţiei (4.44) se poate spune că diferenţa între oricare 2 coeficienţi $c_{1D}(k+h)$ şi $c_{1D}(k)$ depinde doar de valorile eşantioanelor în punctele $k+h$ şi k [90]. Această diferenţă nu depinde de forma funcţiei între cele 2 puncte. De aici rezultă funcţia de transfer a filtrului la ieşirea căruia obţinem coeficienţii:

$$H_{1D}(z) = \frac{z - z^{-1}}{z - z^{-1}} \tag{4.45}$$

Acesta reprezintă de asemenea un filtru cu răspuns infinit la impuls (IIR), cu poli pe cercul unitate.

4.5.4. Algoritmul 1P

Pentru determinarea valorilor derivatelor în noduri se pot utiliza formulele deduse în subcapitolul 4.2.3., dacă pentru funcţia de interpolare se impun condiţii mai tari de continuitate $f \in C^4$ [60].

De această dată coeficienţii vor fi notaţi cu $c_{1P}(k)$.

Algoritmul 1P devine:

- determinarea valorilor primilor 3 coeficienţi:

$$c_{1P}(2) = y(2) - \frac{-(y(0)+y(4)) + 16(y(1)+y(3)) - 30y(2)}{72} = \\ = \frac{(y(0)+y(4)) - 16(y(1)+y(3)) + 54y(2)}{72} \tag{4.46}$$

$$c_{1P}(0) = c_{1P}(2) - \frac{-3y(0) - 10y(1) + 18y(2) + y(4)}{12} \tag{4.47}$$

$$c_{1P}(1) = \frac{6y(1) - c_{1C}(0) - c_{1C}(2)}{4} \tag{4.48}$$

- restul de coeficienţi $c_{1P}(k)$, de la k=2 până la k=N-1, se calculează cu relaţia:

$$c_{1P}(k) = \frac{y(k-3) - 8y(k-2) + 8y(k) - y(k+1)}{6} + c_{1P}(k-2) \tag{4.49}$$

- pentru reconstrucţia semnalului, şirului de coeficienţi $\{c_{1P}(k)\}$ i se aplică filtrul B-spline indirect $B_1^3(z)$;
- pentru interpolarea semnalului cu un factor m se face supraeşantionarea şirului de coeficienţi $c_{1Pm}(i)$, care apoi se aduce la intrarea filtrului B-spline indirect $B_m^3(z)$.

Relaţia (4.49) mai poate fi scrisă şi sub forma [90]:

$$c_{1P}(k+1) - c_{1P}(k-1) = [y(k-2) - 8y(k-1) + 8y(k+1) - y(k+2)]/6 \tag{4.50}$$

De aici rezultă funcţia de transfer a sistemului ce realizează prefiltrarea ca fiind cea din relaţia (4.51). Sistemul reprezintă de asemenea un filtru cu răspuns infinit la impuls, care are polii pe cercul unitate.

$$H_{1P}(z)=\frac{\left(z^{-2}-z^{2}\right)-8\left(z^{-1}+z^{1}\right)}{6\left(z-z^{-1}\right)} \tag{4.51}$$

În continuare se va demonstra convergenţa acestei metode de calcul al coeficienţilor. Se consideră un număr finit de iteraţii succesive [90]. Se pot scrie următoarele relaţii:

$$c_{1P}(3)-c_{1P}(1)=[y(0)-8y(1)+8y(3)-y(4)]/6$$

$$c_{1P}(5)-c_{1P}(3)=[y(2)-8y(3)+8y(5)-y(6)]/6$$

$$c_{1P}(7)-c_{1P}(5)=[y(4)-8y(5)+8y(7)-y(8)]/6$$

...

$$c_{1C}(k)-c_{1C}(k-2)=[y(k-3)-8y(k-2)+8y(k)-y(k+1)]/6$$

Dacă se adună relaţiile din şirul de mai sus, se observă că toţi termenii intermediari se reduc:

$$\Rightarrow \qquad c_{1P}(k)-c_{1P}(1)=[8y(k)-y(k-1)-y(k+1)+8y(1)+y(0)+y(2)]/6$$

Au fost luaţi în considerare doar coeficienţii de indice impar. În general, pentru orice valoare a lui k număr impar, ecuaţia poate fi reformulată astfel:

$$\begin{aligned} c_{1P}(k)-c_{1P}(1)=&\{y(k)-[y(k+1)-2y(k)+y(k-1)]/6\}-\\ &-\{y(1)-[y(2)-2y(1)+y(0)]/6\} \end{aligned} \tag{4.52}$$

Orice diferenţă $c_{1P}(k)$-$c_{1P}(1)$, pentru k impar, nu depinde de valori intermediare, ci doar de eşantioanele $y(k)$, $y(1)$ şi cele din vecinătatea lor. Valorile acestor coeficienţi nu sunt influenţate de eşantioane intermediare. Aceasta înseamnă că funcţia poate fi arbitrară între punctele (k-1) şi 2.

Acelaşi raţionament se poate face şi pentru coeficienţii pari, dacă ne raportăm la $c_{1P}(2)$ în loc de $c_{1P}(1)$. Rezultatul este similar. În acest caz, metoda poate fi generalizată şi utilizată atât pentru funcţii continue, cât şi pentru semnale cu discontinuităţi [90].

Se observă că expresia în paranteze drepte din relaţia (4.52) reprezintă valorile derivatelor numerice de ordinul 2 în punctele k şi respectiv 1 calculate după definiţia lui O. Stănăşilă [87].

4.5.5. Rezultate experimentale comparate

Cei 3 algoritmi prezentaţi mai sus au fost utilizaţi pe rând pentru a realiza operaţia de prefiltrare din procesul de interpolare generalizată, reconstrucţia semnalului (figura 4.1) şi interpolarea cu un factor m=2 (figura 4.2). Au fost implementaţi prin programe în *Matlab*. Secvenţele de program prin care se determină coeficienţii şi câteva rezultate semnificative se regăsesc în *Anexa* 3.

În continuare vor fi prezentate câteva exemple care duc la stabilirea unor concluzii. Valorile din şirul de intrare $\{y(k)\}$, cu k=0, 1, …, N-1 au fost obţinute prin eşantionarea semnalelor $sin(2\pi f_s t)$ şi $cos(2\pi f_s t)$ cu o frecvenţă de eşantionare f_e astfel încât $f_e=Mf_s$ (unde f_s este frecvenţa semnalelor).

Prima dată s-a luat cazul unor semnale sinus şi cosinus cu M=12 eşantioane pe perioadă şi s-au calculat coeficienţii cu *algoritmul 1A*. Majoritatea valorilor interpolate s-au obţinut cu erori de ordinul 10^{-1}. Pentru aceleaşi semnale s-au determinat coeficienţii utilizând *algoritmii 1D* şi *1P*, apoi s-a efectuat interpolarea cu m=2. Erorile de interpolare sunt de ordinul 10^{-2}, respectiv 10^{-3} [90].

Dacă mărim frecvenţa de eşantionare (de exemplu M=120), diferenţele sunt semnificative între erorile de interpolare date de cele 3 metode. Erorile sunt de ordinul 10^{-3} la utilizarea filtrului $H_{1A}(z)$, 10^{-4} pentru $H_{1D}(z)$ şi 10^{-7} dacă prefiltrarea se face cu $H_{1P}(z)$.

În tabelul 4.2. sunt prezentate valori ale erorilor de interpolare în câteva puncte α de pe forma de undă pentru semnalul $y(k)=cos(2\pi k/M)$, cu M=12, respectiv 120.

Tabelul 4.2. Erori de interpolare pentru $y(k)=cos(2\pi k/M)$, prefiltrare cu $H_{1A}(z)$, $H_{1D}(z)$, şi $H_{1P}(z)$

α	M	$H_{1A}(z)$	$H_{1D}(z)$	$H_{1P}(z)$
$\pi/3$	12	$-5{,}50\cdot10^{-2}$	$-8{,}17\cdot10^{-3}$	$2{,}49\cdot10^{-4}$
	120	$-6{,}83\cdot10^{-4}$	$-2{,}26\cdot10^{-4}$	$5\cdot10^{-8}$
$\pi/2$	12	$-3{,}86\cdot10^{-2}$	$-3{,}86\cdot10^{-2}$	$-9{,}97\cdot10^{-4}$
	120	$-9{,}12\cdot10^{-4}$	$-4{,}55\cdot10^{-4}$	$-1{,}5\cdot10^{-7}$
π	12	$-1{,}22\cdot10^{-1}$	$-6{,}63\cdot10^{-2}$	$-2{,}7\cdot10^{-3}$
	120	$-1{,}36\cdot10^{-3}$	$-9{,}12\cdot10^{-4}$	$-4\cdot10^{-7}$

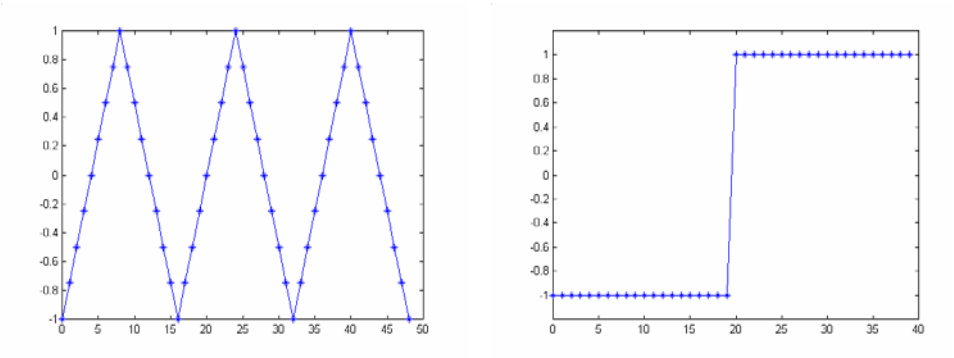
Erorile de interpolare obţinute pentru cel de-al doilea filtru $H_{1D}(z)$ pot fi considerate acceptabile pentru o serie de aplicaţii. Algoritmul este simplu şi necesită efectuarea unui număr relativ mic de operaţii. Rezultate mai bune se obţin în cazul în care se utilizează diferenţa finită centrală a unui polinom ce trece prin 5 puncte $H_{1P}(z)$. Dezavantajul este că în această situaţie creşte numărul de operaţii efectuate [90]. Scăderea erorilor este posibilă prin mărirea volumului de calcule realizate.

Şi în cazul acestor noi algoritmi apar erori la capete datorate atât modului în care se determină valorile coeficienţilor, cât şi metodei de determinare a eşantioanelor interpolate la capete.

Noii algoritmi au fost testaţi şi pe semnale care prezintă discontinuităţi. Astfel, s-au adus la intrare eşantioanele unui semnal triunghiular, apoi ale unui semnal treaptă unitate.

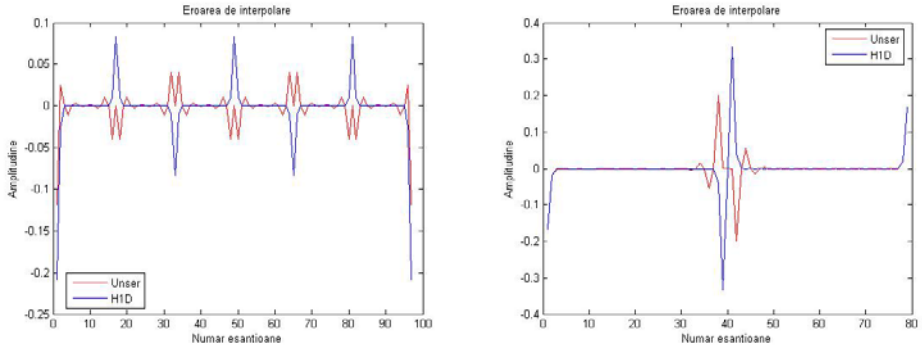
În figura 4.4 sunt prezentate semnalul triunghiular cu 16 eşantioane pe perioadă a), respectiv semnalul treaptă b), pe care sunt evidenţiate punctele în care s-au prelevat eşantioanele de intrare. Pentru fiecare semnal s-au determinat coeficienţii cu ajutorul filtrului $H_{1D}(z)$, respectiv cu filtrul B-spline direct $\left[B_1^3(z)\right]^{-1}$ folosit de M. Unser, apoi s-au calculat eşantioanele semnalelor interpolate cu factor m=2. Erorile de interpolare sunt redate în figura 4.5. Nu se iau în discuţie erorile la capete, care apar în toate cazurile. Erori mari apar în punctele de discontinuitate şi în punctele vecine acestora [52]. În toate celelalte puncte erorile de interpolare sunt

nesemnificative. Se observă că în cazul utilizării *algoritmului 1D*, erorile sunt mai mari decât cele date de algoritmul Unser. Dar, acestea afectează un număr mai mic de eşantioane din vecinătatea punctelor de discontinuitate.



a) semnal triunghiular b) semnal treaptă

Fig. 4.4. Eşantioanele semnalelor cu discontinuităţi.



a) semnal triunghiular b) semnal treaptă

Fig. 4.5. Erorile de interpolare la prefiltrarea cu $H_{1D}(z)$, respectiv cu filtrul B-spline direct.

De exemplu, pentru semnalul treaptă, eroarea absolută maximă (în modul) este $3{,}33 \cdot 10^{-1}$. Aceasta apare între cele 2 puncte unde semnalul sare de la valoarea -1 la 1. Pentru încă 2 puncte învecinate, erorile (în modul) sunt egale cu $4{,}16 \cdot 10^{-2}$. La semnalul triunghiular, eroarea absolută maximă este de $8{,}33 \cdot 10^{-2}$ în punctele de discontinuitate şi $1{,}04 \cdot 10^{-2}$ în vecinătate [52]. În celelalte puncte, erorile de interpolare sunt neglijabile. Toate semnalele pentru care s-au făcut simulări sunt scalate astfel încât valorile lor să se regăsească în intervalul [-1; 1].

Pentru algoritmul Unser, erori apar în mai multe puncte din vecinătatea punctelor de discontinuitate. Oscilaţiile care apar în şirul coeficienţilor determină oscilaţii semnificative în semnalele interpolate [46, 80].

Valorile erorilor de interpolare scad o dată cu creşterea frecvenţei de eşantionare a semnalelor de intrare şi în aceste cazuri. De asemenea, coeficienţii urmăresc variaţia semnalului şi au valori apropiate de acesta, la fel ca şi în cazul algoritmului Unser.

4.6. Algoritmi care utilizează derivata a doua

4.6.1. Consideraţii generale

S-a arătat în subcapitolul 4.3 care sunt legăturile între coeficienţii B-spline şi valorile derivatelor de ordinul 1, respectiv ordinul 2 ale funcţiei B-spline cubică. În orice nod k, coeficientul se poate calcula pe baza valorii funcţiei în acel punct şi a valorii derivatei de ordinul 2 a funcţiei în acelaşi punct:

$$c_s(k) = f(k) - \frac{f''(k)}{6} \tag{4.25}$$

Coeficienţii se determină independent, fără a avea nevoie de valori ale altor coeficienţi calculaţi anterior. Astfel, relaţia de calcul a acestora nu mai este recursivă şi permite determinarea locală a fiecăruia. Valorile coeficienţilor nu sunt influenţate de erori de calcul al altor coeficienţi, ci doar de erorile de determinare a valorilor derivatei a doua în punctele respective.

În acest caz nu mai este nevoie de iniţializarea şirului de coeficienţi la fel ca la algoritmii ce utilizează derivata întâi. Totuşi, la capete, apar probleme datorate modului în care se calculează valoarea derivatei în aceste puncte. De aceea, acolo unde este necesar, coeficienţii vor fi calculaţi utilizând derivate necentrate. De asemene, în aceste cazuri se pot utiliza şi formulele (4.6), (4.7) şi (4.8), ca în cazul algoritmilor prezentaţi în subcapitolul 4.5. Diferenţele între cele 2 cazuri sunt minore şi erorile de determinare a coeficienţilor la capete nu se propagă în restul şirului datorită faptului că fiecare dintre ceilalţi coeficienţi se calculează independent.

Se observă că în relaţia (4.25) apare factorul [$f''(k)/6$], termen ce se regăseşte şi în soluţia matriceală propusă de Gh. Micula pentru determinarea funcţiei de interpolare spline cubică [60].

Şi în această relaţie, pentru determinarea valorilor derivatei a doua în noduri, se pot utiliza fiecare dintre cele 3 metode de derivare numerică descrise în subcapitolul 4.2. Astfel, se dezvoltă 3 algoritmi de interpolare pentru care diferă doar modul de calcul al coeficienţilor: *algoritmul 2A*, *algoritmul 2D* şi *algoritmul 2P*. Fiecare algoritm în parte presupune:

- determinarea coeficienţilor $\{c(k)\}$ prin operaţia de prefiltrare a eşantioanelor semnalului de intrare $\{y(k)\}$, cu k=0, 1,…, N-1. Funcţia de transfer $H_{pf}(z)$ diferă pentru cei 3 algoritmi;
- reconstrucţia semnalului $\{y_r(k)\}$ cu ajutorul filtrul B-spline indirect $B_1^3(z)$;
- pentru interpolarea semnalului $\{y_i(k)\}$ cu un factor m se face supraeşantionarea şirului de coeficienţi $c_m(i)$ (3.48), care apoi se aduce la intrarea filtrului B-spline indirect $B_m^3(z)$.

În continuare se prezintă modalitatea de determinare a coeficienţilor pentru fiecare din cei 3 algoritmi.

4.6.2. Algoritmul 2A

Se ştie că în noduri (punctele k), funcţia $f(k)$ ia valorile eşantioanelor semnalului de intrare $y(k)$. Se utilizează diferenţele finite de ordinul 2 pentru a

calcula valorile *f''*(*k*) [45, 93]. Noii coeficienţi vor fi notaţi: $c_{2A}(k)$. Relaţia (4.25) devine:

$$c_{2A}(k) = y(k) - \frac{[y(k+2) - 2y(k+1) + y(k)]/2}{6}$$

$$\Rightarrow \qquad c_{2A}(k) = \frac{11y(k) - y(k+2) + 2y(k+1)}{12} \tag{4.53}$$

Funcţia de transfer a prefiltrului este [53, 54]:

$$H_{2A}(z) = \frac{11 - \left(z^2 - 2z^1\right)}{12} \tag{4.54}$$

Aceasta este funcţia de transfer a unui filtru cu răspuns finit la impuls (FIR). Filtrele FIR digitale au avantajul că sunt nerecursive, stabile şi foarte simplu de implementat [64, 86].

Aproape toţi coeficienţii, în afară de ultimii 2, pot fi calculaţi cu relaţia (4.53). Pentru ultimii 2 nu avem valorile $y(k+1)$ şi $y(k+2)$ ale semnalului de intrare. De aceea, ei vor fi determinaţi folosind diferenţe finite la stânga (regresive).

4.6.3. Algoritmul 2D

În acest caz, folosim definiţia (4.7) propusă de O. Stănăşilă [87] pentru calculul derivatei de ordinul 2. Notăm coeficienţii cu: $c_{2D}(k)$. În fiecare nod se poate scrie:

$$c_{2D}(k) = y(k) - \frac{y(k+1) - 2y(k) + y(k-1)}{6}$$

$$\Rightarrow \qquad c_{2D}(k) = \frac{8y(k) - y(k+1) - y(k-1)}{6} \tag{4.55}$$

Cu formula (4.55) nu se pot determina prima şi ultima valoare din şirul de coeficienţi. Acestea se calculează cu ajutorul derivatelor necentrate de ordinul 2. Astfel:

$$c_{2D}(0) = y(0) - \frac{y(0) - 2y(1) + y(2)}{6}$$

$$c_{2D}(N-1) = y(N-1) - \frac{y(N-3) - 2y(N-2) + y(N-1)}{6}$$

Funcţia de transfer a sistemului utilizat pentru prefiltrare este [53, 54]:

$$H_{2D}(z) = \frac{8 - \left(z + z^{-1}\right)}{6} \tag{4.56}$$

În acest caz funcţia descrie un filtru FIR simetric.

4.6.4. Algoritmul 2P

Valorile derivatelor se determină cu ajutorul relaţiei (4.17), care defineşte diferenţa finită centrală de ordinul 2 a unui polinom ce trece prin 5 puncte. Pentru coeficienţi folosim notaţia: $c_{2P}(k)$. Relaţia (4.25) devine:

$$c_{2P}(k) = y(k) - \frac{\left[-(y(k-2)+y(k+2)) + 16(y(k-1)+y(k+1)) - 30y(k)\right]/12}{6}$$

$$\Rightarrow \qquad c_{2P}(k) = \frac{102y(k) - 16\left[y(k-1)+y(k+1)\right] + \left[y(k-2)+y(k+2)\right]}{72} \qquad (4.57)$$

Primii şi ultimii 2 coeficienţi din şir se determină cu ajutorul derivatelor de ordinul 2 necentrate.

Funcţia de transfer a sistemului de prefiltrare este [53, 54]:

$$H_{2P}(z) = \frac{102 - 16\left(z + z^{-1}\right) + \left(z^2 + z^{-2}\right)}{72} \qquad (4.58)$$

Şi în acest caz filtrul este cu răspuns finit la impuls, simetric.

Utilizarea filtrelor cu răspuns finit la impuls reprezintă principalul avantaj al acestor 3 algoritmi. Aceste filtre sunt nerecursive, stabile. Fiecare coeficient se calculează doar pe baza valorilor semnalului de intrare, fără a fi utilizaţi coeficienţi determinaţi anteriori. Astfel se evită propagarea erorilor la calculul coeficienţilor. Orice oscilaţie apărută în semnalul de intrare la un moment dat nu se propagă decât pentru un anumit număr finit de termeni (3 sau 5, în funcţie de metoda de determinare a valorilor derivatei). Pentru *algoritmul 2D* şi *algoritmul 2P*, filtrele sunt simetrice, ceea ce înseamnă că sunt cu fază liniară. Proprietatea de fază liniară presupune că toate componentele spectrale ale semnalului de intrare sunt întârziate la fel. Proprietatea de simetrie permite şi reducerea numărului de operaţii necesare implementării acestor filtre [64, 86].

4.6.5. Rezultate experimentale comparate

Şi algoritmii de interpolare care utilizează derivata a doua pentru calculul coeficienţilor au fost implementaţi prin programe în *Matlab*. Ca şi valori de intrare $\{y(k)\}$, cu k=0, 1, …, N-1, au fost utilizate de asemenea eşantioanele semnalelor $sin(2\pi k/M)$ şi $cos(2\pi k/M)$, cu M=12, respectiv 120 şi N ales astfel încât să corespundă la 3 perioade din semnal. În acest fel se pot face comparaţii între diverşii algoritmi prezentaţi. Secvenţele de program prin care se determină coeficienţii şi unele rezultate semnificative sunt redate în *Anexa 4*. Câteva rezultate au fost sintetizate şi vor fi prezentate în continuare.

Coeficienţii urmăresc variaţia semnalului de intrare şi au valori apropiate de acesta, ca şi în celelalte cazuri studiate. La fel ca în celelalte cazuri, erorile la capete sunt mai mari decât erorile pentru restul valorilor interpolate.

Pentru $y(k)=cos(2\pi k/M)$, cu $M=12$ şi $k=0, 1, \ldots, 36$ (eşantioane ce corespund la 3 perioade din semnal), s-a realizat interpolarea cu un factor de interpolare $m=2$ utilizând *algoritmul 2A*. Erorile de interpolare sunt prezentate în figura 4.6. Excluzând valorile de la capete, erorile sunt de maxim $2{,}8\cdot10^{-2}$.

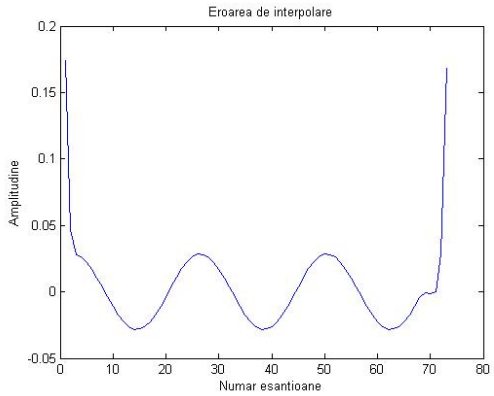


Fig. 4.6. Erori pentru $cos(2\pi k/12)$ la interpolarea cu *algoritmul 1A*.

Interesează în mod special rezultatele obţinute în cazurile în care sunt utilizate $H_{2D}(z)$ şi $H_{2P}(z)$, deoarece acestea sunt filtre cu răspuns finit la impuls simetrice. În tabelele 4.3, respectiv 4.4, sunt prezentate rezultatele pentru semnalul sinusoidal, în cazul în care s-a folosit filtrul $H_{2D}(z)$, respectiv $H_{2P}(z)$ [53].

Tabelul 4.3. Rezultate pentru $y(k)=sin(2\pi k/M)$, prefiltrare cu $H_{2D}(z)$.

α	M	c_{2B}	e_{in}
$\pi/3$	12	0,9047	$3{,}08\cdot10^{-3}$
	120	0,8664	$1{,}8\cdot10^{-7}$
$\pi/2$	12	1,0446	$1{,}99\cdot10^{-3}$
	120	1,0004	$2\cdot10^{-7}$
π	12	0	0
	120	0	0

Tabelul 4.4. Rezultate pentru $y(k)=sin(2\pi k/M)$, prefiltrare cu $H_{2P}(z)$.

α	M	c_{2C}	e_{in}
$\pi/3$	12	0,9055	$6{,}25\cdot10^{-4}$
	120	0,8664	$9\cdot10^{-8}$
$\pi/2$	12	1,0456	$1{,}04\cdot10^{-3}$
	120	1,0004	$1\cdot10^{-7}$
π	12	0	0
	120	0	0

În fiecare caz sunt reproduse valorile coeficienţilor şi ale erorilor de interpolare obţinute în câteva puncte α distincte de pe forma de undă, unde $\alpha=2\pi k/M$. Se observă că erorile de interpolare au valori mai mici în cazul în care s-a folosit derivata centrată a unui polinom ce trece prin 5 puncte. Dar diferenţele nu sunt mari, de ordinul 10^{-3}, nici chiar pentru semnalele eşantionate cu o frecvenţă mai mică (pentru M=12). Pentru semnalele cu mai multe eşantioane pe perioadă (M=120) erorile de interpolare sunt neglijabile.

Câteva valori obţinute pentru semnalul cosinus sunt redate în tabelele 4.5 (prefiltrare cu $H_{2D}(z)$) şi respectiv 4.6 (pentru $H_{2P}(z)$). Erorile de interpolare sunt de acelaşi ordin ca şi pentru semnalul sinus.

Tabelul 4.5. Rezultate pentru $y(k)=cos(2\pi k/M)$, prefiltrare cu $H_{2D}(z)$.

α	M	c_{2B}	e_{in}
$\pi/3$	12	0,5223	$-3{,}64\cdot10^{-4}$
	120	0,5002	$1\cdot10^{-7}$
$\pi/2$	12	0	0
	120	0	0
π	12	-1,0446	$-1{,}99\cdot10^{-3}$
	120	-1,0004	$-2\cdot10^{-7}$

În cazul acestor semnale s-ar putea spune că pentru condiţii mai tari de continuitate pentru funcţia care aproximează semnalul de intrare nu se obţin rezultate mai bune la interpolare. Aceasta se datorează şi faptului că funcţiile *sin* şi *cos* sunt continue şi cu derivatele continue. Pentru algoritmii implementaţi am presupus că orice funcţie de intrare este aproximată printr-o funcţie spline cubică.

Tabelul 4.6. Rezultate pentru $y(k)=cos(2\pi k/M)$, prefiltrare cu $H_{2P}(z)$.

α	M	c_{2C}	e_{in}
$\pi/3$	12	0,5228	$4{,}57\cdot10^{-4}$
	120	0,5002	$5\cdot10^{-8}$
$\pi/2$	12	0	0
	120	0	0
π	12	-1,0456	$-1{,}04\cdot10^{-3}$
	120	-1,0004	$-1\cdot10^{-7}$

Comparăm aceste rezultate cu cele din tabelul 4.2, obţinute cu algoritmul ce utilizează valorile derivatei întâi. Erorile de interpolare sunt în aceeaşi plajă ca şi în cazul folosirii filtrului $H_{1P}(z)$.

În figura 4.7 sunt prezentate comparativ erorile obţinute la interpolarea cu un factor m=2 a semnalului $cos(2\pi k/12)$ pentru cazurile în care interpolarea s-a făcut cu *algoritmul 1P* (roşu), *algoritmul 2D* (verde) şi respectiv *algoritmul Unser* (albastru). Nu au fost reprezentate erorile la capete. Se observă că erorile obţinute la interpolarea cu *algoritmul 1P* sunt cam de acelaşi ordin de mărime cu cele obţinute cu *algoritmul 2D*, dar mai mari decât cele date de *algoritmul Unser*. *Algoritmul 2D*, faţă de *1P*, are avantajul că fiecare coeficient este calculat

independent de ceilalţi şi la determinarea acestuia se utilizează un număr mai mic de termeni. De aici rezultă mai puţine operaţii de efectuat la implementarea algoritmului pe un sistem numeric de prelucrare.

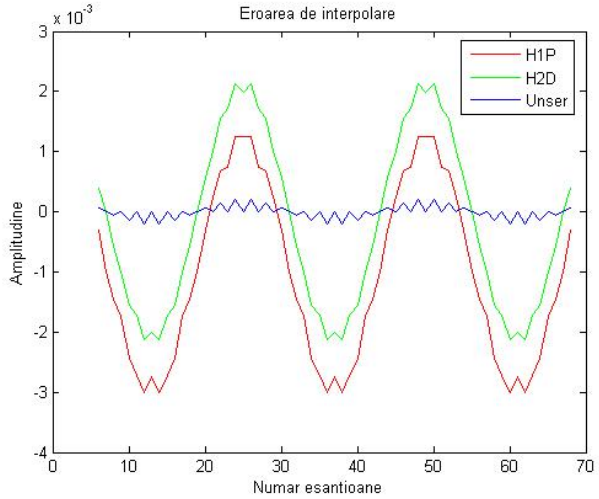


Fig. 4.7. Erori pentru $cos(2\pi k/12)$ la interpolarea cu *algoritmii 1P*, *2D* şi *Unser*.

Se compară erorile de interpolare obţinute la prelucrarea aceluiaşi semnal $cos(2\pi k/12)$ cu algoritmii 2P, 2D şi Unser. Rezultatele (fără erorile la capete) sunt redate în figura 4.8.

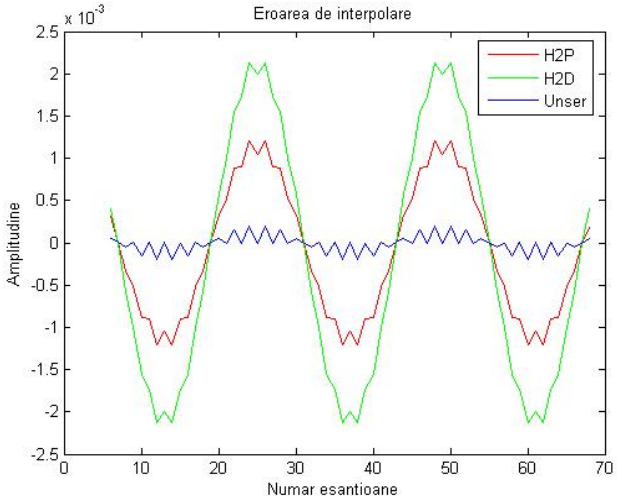


Fig. 4.8. Erori pentru $cos(2\pi k/12)$ la interpolarea cu *algoritmii 2P*, *2D* şi *Unser*.

Se observă că pentru cei 2 algoritmi ce utilizează valorile derivatei a doua la calcularea coeficienţilor erorile maxime sunt de ordinul 10^{-3}, diferind ca valoare. În acest caz, algoritmul 2P nu oferă rezultate cu mult mai bune la interpolare.

La creşterea de 10 ori a numărului de eşantioane pe perioadă (corespunzător creşterii de 10 ori a frecvenţei la care a fost eşantionat semnalul de intrare), pentru toţi algoritmii discutaţi, la interpolare s-au obţinut erori care pot fi considerate zero. Acest fapt reiese şi din figura 4.9 în care sunt prezentate erorile de interpolare cu factorul m=2 a semnalului $cos(2\pi k/120)$ pentru 3 dintre aceşti algoritmi.

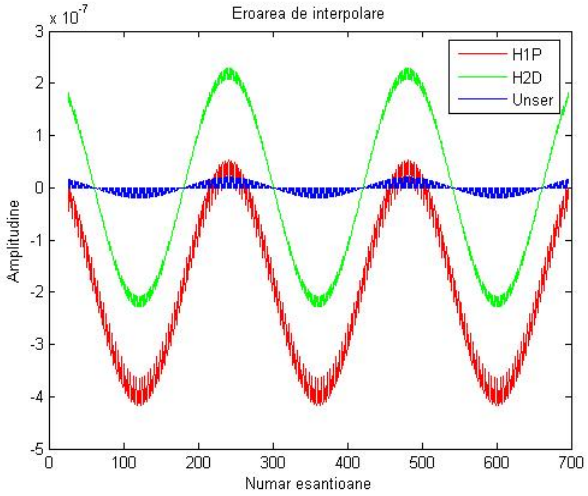


Fig. 4.9. Erori pentru $cos(2\pi k/120)$ la interpolarea cu *algoritmii 1P*, *2D* şi *Unser*.

Pentru toate exemplele studiate, în noduri, *algoritmul Unser* realizează interpolarea exactă $f(k)=y(k)$. Algoritmii propuşi în acest capitol oferă o aproximare a semnalului de intrare. Aproximarea este cu atât mai bună cu cât creşte frecvenţa la care au fost eşantionate semnalele de intrare.

Vom prezenta în continuare şi câteva rezultate obţinute la interpolarea unor semnale de intrare care prezintă discontinuităţi. Sunt luate eşantioane din semnalele triunghi şi treaptă (figura 4.4), la fel ca şi pentru algoritmii prezentaţi în subcapitolul 4.5. Şi aici se vede cum coeficienţii urmăresc forma semnalului şi au valori apropiate de acesta, iar erorile de interpolare descresc odată cu creşterea frecvenţei de eşantionare [54].

Doar un număr mic de valori din jurul punctelor de discontinuitate este influenţat de erori semnificative şi în cazul utilizării acestor algoritmi. Astfel, prin faptul că nu se propagă erori de calcul în şirul coeficienţilor, se minimizează sursele de erori pentru şirul de date interpolate. Pentru semnalul triunghiular cu M=16 eşantioane pe perioadă, valorile maxime ale erorilor de interpolare sunt: $2{,}95\cdot10^{-2}$ la utilizarea filtrului $H_{2D}(z)$ şi respectiv $3{,}26\cdot10^{-2}$ pentru $H_{2P}(z)$ [54]. La utilizarea filtrului B-spline direct din algoritmul rapid de interpolare cu funcţii B-spline cubice (*algoritmul Unser*), valoarea maximă, în modul, a erorilor este $3{,}96\cdot10^{-2}$. Şi în aceste cazuri am ignorat valorile erorilor la capete, care au ca principală cauză modul în care se calculează eşantioanele semnalului interpolat.

În figura 4.10 sunt prezentate erorile pentru semnalul triunghiular amintit la interpolarea cu *algoritmul 2D*, respectiv cu *algoritmul Unser*.

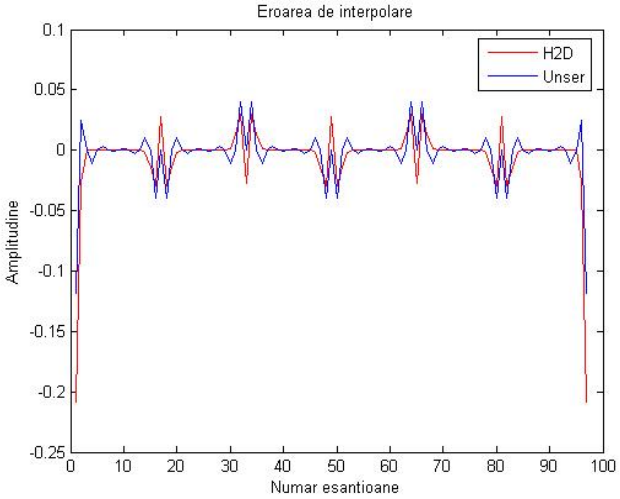


Fig. 4.10. Erori pentru semnal triunghiular la interpolarea cu *algoritmii 2D* şi *Unser*.

Se observă că în ambele cazuri erorile de interpolare au valori apropiate dar, în vecinătatea discontinuităţilor un număr mai mic de puncte sunt afectate de erori semnificative la interpolarea cu *algoritmul 2D*.

Pentru semnalul treaptă, erorile din jurul punctelor de discontinuitate sunt de maxim $1{,}66 \cdot 10^{-1}$ la folosirea primului filtru şi respectiv $1{,}35 \cdot 10^{-1}$ pentru celălalt, faţă de $3{,}33 \cdot 10^{-1}$ în cazul filtrului B-spline direct. În figura 4.11. sunt prezentate erorile de interpolare în cazul utilizării *algoritmilor 2D* şi respectiv *Unser*.

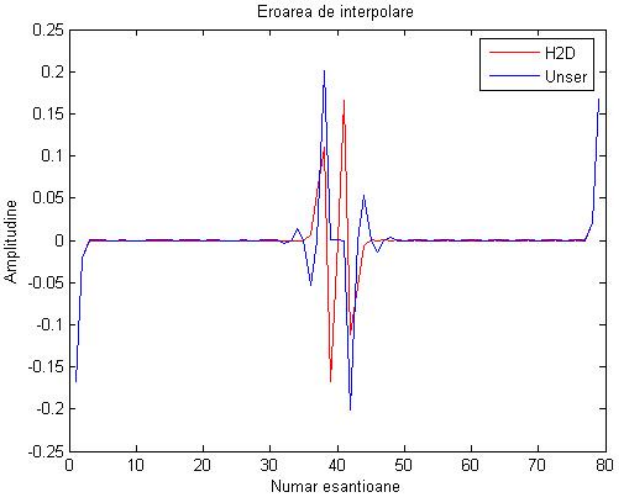


Fig. 4.11. Erori pentru semnal treaptă la interpolarea cu *algoritmii 2D* şi *Unser*.

Şi pentru acest exemplu erorile afectează mai puţine eşantioane din jurul punctelor de discontinuitate la interpolarea cu *algoritmul 2D*, faţă de interpolarea cu algoritmul Unser. Valoarea maximă a erorilor de interpolare este chiar mai mică la prefiltrarea cu $H_{2D}(z)$. Pentru semnalele care prezintă puncte de discontinuitate se poate spune că *algoritmul 2D* oferă rezultate mai bune la interpolare decât algoritmul rapid de interpolare cu funcţii B-spline cubice (algoritmul Unser).

La interpolarea cu *algoritmul 2P* a celor 2 semnale discontinue aduse în discuţie, erorile sunt aproximativ la fel ca la prelucrarea cu *algoritmul 2D*. Şi din studiul acestor exemple se poate spune că pentru condiţii mai stricte de continuitate pentru funcţia care aproximează semnalul de intrare, nu se obţin rezultate cu mult mai bune [54].

Reamintim faptul că în programele *Matlab* variabilele au fost declarate de tip *double*, deci s-a lucrat cu precizie extinsă, iar valorile mai mici de 10^{-8} sunt considerate nule.

Pentru algoritmul *2D* s-a analizat variaţia cu frecvenţa de eşantionare a erorii maxime de interpolare pentru m=2. La intrarea sistemului s-au adus pe rând eşantioanele corespunzătoare semnalului $sin(2\pi k/M)$ pentru mai multe valori ale factorului M (5, 6, 8, 10, 12, ...,240). În figura 4.12 este prezentată eroarea maximă de interpolare în funcţie de numărul de eşantioane pe perioadă (raportul frecvenţă de eşantionare f_e/frecvenţă semnal f_s) [56]. Şi de această dată s-au ignorat erorile la capete.

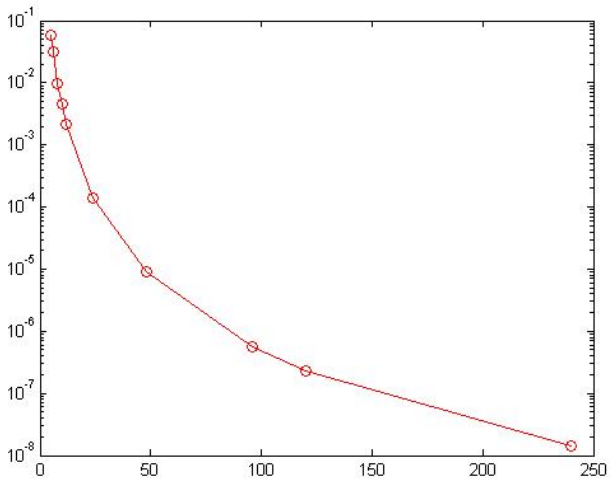


Fig. 4.12. Variaţia cu M a erorii maxime de interpolare pentru *algoritmul 2D*.

Se observă că valorile erorilor scad logaritmic, la fel ca pentru algoritmul Unser (figura 3.19.). Cu cât creşte frecvenţa de eşantionare la care s-au obţinut datele de intrare, cu atât valorile coeficienţilor sunt mai apropiate de valorile semnalului şi aproximarea mai bună.

4.7. Concluzii

În acest capitol s-au prezentat noi algoritmi de interpolare spline cubică dezvoltaţi de autoarea acestei teze. Punctul de plecare îl reprezintă *algoritmul rapid de interpolare spline cubică* realizat de M. Unser şi colaboratorii [100, 108, 112]. Noutatea este dată de modalităţile de determinare a coeficienţilor B-spline în procesul de prefiltrare. Din *algoritmul rapid de interpolare spline cubică* se păstrează partea a doua, cea de interpolare, respectiv de reconstrucţie a semnalului. Pentru primul pas (prefiltrarea) se înlocuieşte filtrul B-spline direct cu noi filtre. În cazul noilor algoritmi, eşantioanele semnalului sunt aduse la intrarea unor sisteme ale căror funcţii de transfer au fost determinate cu ajutorul valorilor derivatelor întâi şi a doua ale funcţiei spline cubice de aproximare în noduri. Interpolarea, respectiv reconstrucţia semnalului se fac prin filtrare numerică, utilizând filtrele B-spline indirecte.

La determinarea funcţiei spline cubice de interpolare prin metoda matriceală se folosesc şi valorile derivatelor funcţiei în noduri. Plecând de la această observaţie, s-au analizat proprietăţile funcţiilor spline cubice, ale derivatelor acestora de ordinul 1 şi 2 şi relaţiile dintre acestea şi coeficienţii B-spline. De aici s-au dezvoltat noile metode de determinare a coeficienţilor prin filtrare numerică. Procesul se poate realiza prin intermediul a 3 filtre IIR (descrise în subcapitolul 4.5) sau a 3 filtre FIR (subcapitolul 4.6), dintre care 2 sunt simetrice. Filtrele FIR au avantajul că sunt nerecursive, stabile şi foarte simplu de implementat în programele de prelucrare numerică a datelor. Se descriu prin operaţii simple de adunare şi înmulţire care pot fi realizate uşor şi eficient cu un procesor numeric de semnal.

În cazul clasic, *algoritmului rapid de interpolare spline cubică*, pentru calcularea coeficienţilor este necesar ca întregul şir de date de intrare să fie cunoscut încă de la început. Acest fapt nu permite utilizarea algoritmului în aplicaţii de prelucrare a semnalelor în timp real. În schimb, noii algoritmi dezvoltaţi pot fi utilizaţi în aplicaţii de prelucrare numerică în timp real. În orice moment, este nevoie doar de câteva valori din semnalul de la intrare pentru a calcula coeficientul corespunzător şi respectiv eşantionul curent de la ieşire. De exemplu, pentru algoritmii descrişi în subcapitolul 4.6, în orice nod k, pentru determinarea coeficientului $c(k)$ sunt necesare doar un număr mic de eşantioane de intrare din jurul acestui punct (3 sau 5). Aceşti algoritmi permit determinarea locală a coeficienţilor. Nici o altă metodă din cele întâlnite în literatura de specialitate nu permite acest lucru. Prin faptul că fiecare coeficient poate fi calculat independent de valorile altora se evită propagarea erorilor de determinare a coeficienţilor.

Filtrele FIR simetrice folosite în *algoritmii 2D* şi *2P* au în plus proprietatea de fază liniară (toate componentele spectrale ale semnalului de intrare sunt întârziate la fel) şi pot fi implementate pe sisteme cu procesoare de semnal printr-un număr redus de operaţii.

Noile modalităţi de determinare a coeficienţilor nu necesită prelungirea semnalelor de intrare. Astfel, sunt reduse erorile la capete datorate iniţializării şirului de coeficienţi. Rămân însă cele ce provin din modalitatea de calcul a valorilor interpolate pe baza coeficienţilor.

Algoritmii au fost implementaţi în programe în *Matlab* şi utilizaţi pentru interpolarea câtorva tipuri de semnale des întâlnite în aplicaţii. Din analiza rezultatelor s-au putut formula câteva concluzii.

În cazul *algoritmilor 1P* şi *2P*, pentru date de intrare provenite din eşantionarea semnalelor sinus şi cosinus, erorile de interpolare nu sunt cu mult mai mici faţă de cele obţinute cu *algoritmii 1D* şi respectiv *2D*, însă, pentru realizarea

primilor doi algoritmi este necesară creşterea numărului de operaţii faţă de implementarea celorlaltor doi. Diferenţele mici între rezultate pot să nu justifice timpul de calcul suplimentar şi eventualele întârzieri.

La intrarea sistemului s-au adus şiruri de date obţinute prin eşantionarea la frecvenţe diferite a aceluiaşi semnal. Coeficienţii determinaţi tind tot mai mult spre valorile semnalului odată cu creşterea frecvenţei de eşantionare (creşterea numărului de eşantioane pe perioadă). Cu cât frecvenţa de eşantionare este mai mare, cu atât mai apropiate sunt aceste valori. În consecinţă, erorile de interpolare scad odată cu creşterea numărului de eşantioane pe perioadă.

Pentru semnalele triunghiular şi treaptă (semnale care prezintă discontinuităţi), la interpolarea cu noii algoritmi se observă că, faţă de interpolarea cu *algoritmul rapid de interpolare spline cubică*, un număr mai mic de eşantioane din vecinătatea punctelor de discontinuitate sunt influenţate de erori mai mari decât erorile pentru restul eşantioanelor. Pentru aceste semnale, interpolarea cu *algoritmul 2D* oferă rezultate mai bune decât interpolarea cu *algoritmul Unser*.

Noii algoritmi pot fi utilizaţi pentru orice tip de semnal. În viitor se vor analiza şi rezultatele interpolării altor tipuri de semnale cu aceşti algoritmi.

Şi în cazurile prezentate în acest capitol, la capetele semnalelor interpolate sau reconstruite apar erori ce au valori mult mai mari decât pentru restul eşantioanelor din şir (denumite *erori la capete*). O metodă de reducere a acestor erori va fi prezentată în capitolul următor, împreună cu câteva modalităţi de îmbunătăţire a algoritmilor de interpolare prezentaţi.

5. ÎMBUNĂTĂŢIREA ALGORITMILOR DE INTERPOLARE SPLINE CUBICĂ CE UTILIZEAZĂ VALORILE DERIVATELOR FUNCŢIEI ÎN NODURI

5.1. Introducere

În continuare se vor prezenta câteva modalităţi de îmbunătăţire a algoritmilor de interpolare descrişi în capitolul 4. Se urmăreşte reducerea erorilor de interpolare a semnalelor prin completarea metodelor de determinare a coeficienţilor B-spline şi micşorarea erorilor la capete.

După cum s-a arătat anterior, pentru toţi algoritmii implementaţi s-a observat apariţia unor erori la capete cu valori mai mari decât erorile pentru restul şirului de eşantioane. În cazul *algoritmului rapid de interpolare spline cubică*, aceste erori afectează un număr mare de eşantioane. Acesta variază în funcţie de numărul de eşantioane pe perioadă (M) al semnalului de intrare. Cu cât semnalul este eşantionat la o frecvenţă mai mare, cu atât mai multe eşantioane sunt afectate de erorile la capete. În cazul noilor algoritmi dezvoltaţi problema a fost parţial soluţionată prin modul de calcul al coeficienţilor la capetele şirului, fiind afectate un număr mult mai mic de eşantioane. Însă, a rămas, ca sursă a acestor erori, modalitatea de determinare a eşantioanelor interpolate din valorile coeficienţilor. Metoda aleasă pentru rezolvarea acestei probleme este de a prelungi şirul de coeficienţi la ambele capete.

Pentru determinarea coeficienţilor B-spline noii algoritmi folosesc şi valorile derivatelor funcţiei spline în noduri, valori ce sunt calculate pe baza eşantioanelor de intrare prin filtrare numerică. Pentru determinarea acestor valori se poate folosi şi derivarea analogică a semnalului de intrare. Astfel, semnalul iniţial şi semnalul derivat, sunt eşantionate fiecare în parte şi apoi sunt aduse la intrarea sistemului de prelucrare. Astfel, cele 2 şiruri de eşantioane pot fi folosite direct pentru determinarea coeficienţilor B-spline. Însă, pentru implementarea acestei metode trebuie avute în vedere alte probleme, legate de dimensionarea şi realizarea circuitului de derivare şi de achiziţia a două semnale simultan.

În toate exemplele prezentate anterior s-au determinat coeficienţii B-spline, apoi s-au calculat eşantioanele semnalelor reconstruite sau interpolate cu un factor de interpolare $m=2$. Vor fi prezentate în continuare şi 2 modalităţi prin care se poate face interpolarea semnalelor cu un factor $m>2$ şi vor fi analizate câteva rezultate experimentale.

5.2. Utilizarea relaţiei din *algoritmul zero*

5.2.1. Relaţia de calcul din *algoritmul zero*

Pentru metodele care folosesc valorile derivatei întâi în noduri la determinarea coeficienţilor se pot identifica 2 surse principale care duc la apariţia erorilor de interpolare. Una dintre ele este modalitatea de calcul a valorilor derivatei

întâi în noduri. Cealaltă este dată de faptul că fiecare coeficient este calculat pe baza valorilor funcţiei şi a valorilor unor coeficienţi determinaţi anterior. Astfel, erorile de determinare a oricărui coeficient se pot propaga în tot şirul. S-au căutat unele metode de a micşora influenţa acestor erori. Aceste metode trebuie să fie cât mai simple şi uşor de implementat. De asemenea, trebuie ca ele să presupună introducerea unui număr mic de operaţii suplimentare.

Algoritmul zero (analizat în subcapitolul 4.4.) prezintă o variantă de calcul a coeficienţilor care poate oferi rezultate bune dacă este folosită pentru un şir cu număr mic de eşantioane de intrare. Relaţia (4.31) poate fi reformulată astfel:

$$c_0(k) = \frac{6y(k) - c_0(k-1) - c_0(k+1)}{4} \tag{5.1}$$

Această formulă va fi folosită adiţional la algoritmii de determinare a coeficienţilor care utilizează valorile derivatei întâi în noduri. Vor fi prezentate în continuare 2 variante dezvoltate pentru îmbunătăţirea metodelor de determinare a coeficienţilor B-spline.

5.2.2. Calculul intercalat al coeficienţilor

Pentru fiecare dintre cei 3 algoritmi descrişi în subcapitolul 4.5 se calculează coeficientul curent $c_1(k)$ în funcţie de valorile câtorva eşantioane de intrare şi a coeficientului întârziat cu 2 tacte $c_1(k-2)$. În acest fel se determină valorile coeficienţilor din 2 în 2 puncte. După stabilirea primelor 3 valori (condiţiile iniţiale), se poate spune că se calculează independent 2 şiruri de coeficienţi: cei de indice par şi cei de indice impar.

Pornind de la această observaţie, se stabileşte următoarea metodă (denumită metoda de calcul intercalat al coeficienţilor):

- un şir de coeficienţi se calculează cu formula cu derivata întâi corespunzătoare;
- celălalt şir se calculează cu formula de la *algoritmul zero*.

Se aplică acest principiu pentru algoritmul care utilizează determinarea valorilor derivatelor în noduri prin formulele lui O. Stănăşilă (*algoritmul 1D*). Astfel, *algoritmul 1D intercalat* presupune parcurgerea următorilor paşi [50]:

- se calculează coeficienţii iniţiali la fel ca în cazul *algoritmului 1D*:

$$c_{1Di}(2) = y(2) - [y(3) - 2y(2) + y(1)]/6$$

$$c_{1Di}(0) = c_{1Di}(2) - y(2) + y(0)$$

- fiecare coeficient de indice par se determină la fel ca în cazul *algoritmului 1D*:

$$c_{1Di}(k) = y(k) - y(k-2) + c_{1Di}(k-2) \tag{5.2}$$

- fiecare coeficient de indice impar se calculează pe baza formulei de la *algoritmul zero* (5.1):

$$c_{1Di}(k-1) = [6y(k-1) - c_{1Di}(k-2) - c_{1Di}(k)]/4 \ , \ k=2, 4, 6,... \tag{5.3}$$

- se supraeşantionează şirul de coeficienţi şi se calculează eşantioanele semnalului interpolat.

În acelaşi mod se poate proceda şi pentru *algoritmii 1A* şi *1P*. Este posibil ca pentru coeficienţii de indice impar să se reducă erorile de determinare. Însă, toţi coeficienţii de indice par sunt însoţiţi de aceleaşi erori de calcul ca şi cei determinaţi cu algoritmii descrişi în subcapitolul 4.5, deoarece sunt calculaţi prin aceleaşi iteraţii, pe baza aceloraşi valori.

O îmbunătăţire a modului de determinare a coeficienţilor duce la rezultate mai bune şi pentru semnalul interpolat. Însă, această metodă poate să nu ofere rezultate mai bune pentru toţi termenii implicaţi. Nu poate fi considerată o variantă viabilă de îmbunătăţire a algoritmilor vizaţi.

5.2.3. Introducerea unui pas suplimentar

După cum s-a văzut în paragraful anterior, folosirea formulei din *algoritmul zero* poate aduce unele îmbunătăţiri. Aceasta poate fi introdusă ca pas suplimentar la calculul coeficienţilor cu algoritmii descrişi în subcapitolul 4.5. Vom lua în continuare acelaşi caz, cel al *algoritmului 1D*. Noul algoritm, *algoritmul 1D cu pas suplimentar*, va fi:

- se calculează coeficienţii iniţiali la fel ca pentru *algoritmul 1D*;
- pentru $k>2$, fiecare coeficient $c_{1D}(k)$ se determină la fel ca în cazul algoritmului *1D*:

$$c_{1D}(k) = y(k) - y(k-2) + c_{1D}(k-2)$$

- pe baza acestora se calculează un set suplimentar de coeficienţi:

$$c_{1Ds}(k-1) = [6y(k-1) - c_{1D}(k-2) - c_{1D}(k)]/4 \qquad (5.4)$$

- eşantioanele interpolate se calculează din setul suplimentar de coeficienţi $c_{1Ds}(k)$.

Pasul suplimentar este echivalent cu introducerea a unui filtru suplimentar în etapa de prefiltrare. Procesul de interpolare este redat schematic în figura 5.1.

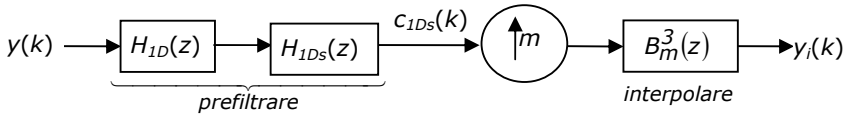


Fig.5.1. Algoritmul 1D cu pas suplimentar.

Prin această metodă, pentru noii coeficienţi se pot diminua erorile de determinare. Fiind diminuate oscilaţiile din şirul de coeficienţi, atunci şi erorile de interpolare vor fi mai mici.

Acest pas adăugat la modalitatea de determinare a coeficienţilor se poate introduce şi pentru algoritmii care utilizează valorile derivatei a doua în noduri (descrişi în subcapitolul 4.6).

Dar, un pas suplimentar înseamnă introducerea în plus a unor operaţii de adunare şi înmulţire, deci creşterea volumului şi a timpului de calcul. Acesta este un

aspect important dacă se doreşte implementarea algoritmului pe un sistem cu procesor numeric de semnal. Pentru aplicaţiile în timp real, apariţia unui filtru în plus înseamnă introducerea de întârzieri suplimentare pentru semnalul de la ieşirea sistemului.

5.2.4. Rezultate experimentale

Algoritmii intercalaţi şi cei cu pas suplimentar au fost implementaţi în Matlab şi testaţi pentru aceleaşi semnale folosite şi în cazurile anterioare. Secvenţele de program corespunzătoare pentru calcularea coeficienţilor şi o parte dintre rezultatele interpolării unor semnale sunt prezentate în *Anexa 5*.

Drept exemplu, pentru $\{y(k)\}$, cu k=0, 1, …, N-1, au fost utilizate tot eşantioanele semnalelor $sin(2\pi k/M)$ şi $cos(2\pi k/M)$, cu M=12, respectiv 120 şi N ales astfel încât: N=3M. S-a realizat reconstrucţia semnalului pentru cazul în care coeficienţii sunt determinaţi prin calcul intercalat (*algoritmul 1D intercalat*). Erorile de interpolare obţinute au fost comparate cu cele determinate cu *algoritmul 1D*. Pentru punctele în care coeficienţii s-au determinat cu formula (5.2), erorile sunt de ordinul 10^{-2} (pentru M=12) şi 10^{-4} (M=120), acelaşi ordin de mărime ca şi în cazul *algoritmului 1D*. În celelalte puncte, în care coeficienţii s-au calculat intercalat (cu formula (5.3)), erorile de interpolare sunt neglijabile.

La interpolarea cu factor m=2, erorile de interpolare au acelaşi ordin de mărime (10^{-2} pentru M=12) şi în punctele intermediare, valorile fiind aproximativ de 2 ori mai mici decât în cazul algoritmului 1D. Excepţie fac doar nodurile în care coeficienţii s-au calculat intercalat, unde valorile erorilor sunt neglijabile [50].

O îmbunătăţire semnificativă se obţine doar în unele puncte, ceea ce duce la concluzia că această metodă poate să nu fi eficientă.

Pentru aceleaşi semnale s-a realizat interpolarea şi cu algoritmii în care determinarea coeficienţilor se face utilizând valorile derivatei întâi în noduri, la care s-a adăugat pasul suplimentar.

În tabelul 5.1 sunt prezentate erorile de interpolare în câteva puncte α de pe forma de undă pentru semnalul $y(k)=sin(\alpha)$, cu $\alpha=2\pi k/M$, unde k=0, 1, …, N-1, pentru 2 cazuri: coeficienţii calculaţi cu *algoritmul 1D*, respectiv *algoritmul 1D cu pas suplimentar* [50]. Se observă că în cel de-al doilea caz erorile de interpolare se reduc faţă de primul caz. Acelaşi algoritm a fost aplicat şi semnalului $cos(2\pi k/M)$, rezultatele fiind similare.

Tabelul 5.1. Erori de interpolare pentru $y(k)=sin(\alpha)$: comparaţie *1D – 1D cu pas suplimentar*.

α	M	*algoritm 1D*	*1D cu pas suplimentar*
0	12	$3{,}05\cdot10^{-2}$	$1{,}11\cdot10^{-2}$
	120	$3{,}58\cdot10^{-5}$	$1{,}19\cdot10^{-5}$
$\pi/3$	12	$8{,}17\cdot10^{-3}$	$5{,}58\cdot10^{-3}$
	120	$3{,}59\cdot10^{-4}$	$1{,}85\cdot10^{-4}$
$\pi/2$	12	$2{,}23\cdot10^{-2}$	$4{,}08\cdot10^{-3}$
	120	$4{,}20\cdot10^{-4}$	$2{,}16\cdot10^{-4}$

Metodele de intercalare şi adăugare a unui pas suplimentar la etapa de prefiltrare pot fi aplicate şi pentru *algoritmul 1P*. În figura 5.2 sunt comparate erorile de interpolare a semnalului $y(k)=cos(2\pi k/12)$ la prelucrarea cu *algoritmii 1P*,

1P intercalat şi *1P cu pas suplimentar*. La folosirea celui de-al doilea algoritm, erorile scad foarte mult doar în unele noduri, la fel ca şi în cazul *algoritmului 1D intercalat*. Pentru interpolarea cu algoritmii cu pas suplimentar, erorile sunt diminuate în aceeaşi proporţie pentru toate eşantioanele şi sunt inversate.

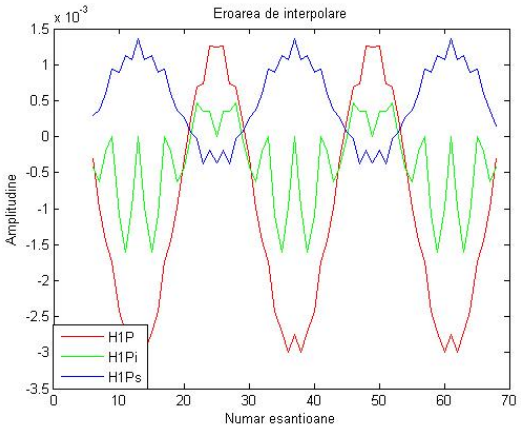
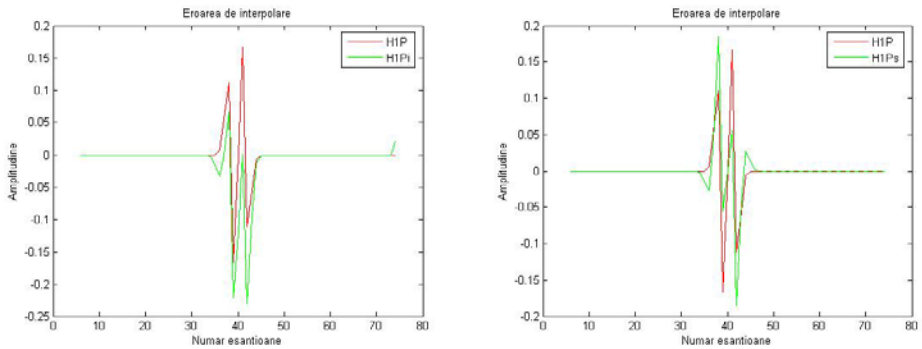


Fig.5.2. Erori pentru $cos(2\pi k/12)$ la interpolarea cu *algoritmii 1P*, *1Pi* şi *1Ps*.

Aceeaşi algoritmi au fost utilizaţi şi pentru interpolarea semnalului treaptă ce a mai fost folosit şi în unele exemple anterioare. Din figura 5.3 se poate observa că pentru unele dintre eşantioanele din vecinătatea punctelor de discontinuitate erorile de interpolare sunt mai mari în cazul *algoritmilor 1P intercalat* (figura 5.3.a)) şi respectiv *1P cu pas suplimentar* (figura 5.3.b)), faţă de prelucrarea cu algoritmul iniţial *1P*.



a) comparaţie *1P* cu *1P intercalat* b) comparaţie *1P* cu *1P cu pas suplimentar*

Fig.5.3. Erori la interpolarea semnalului treaptă cu *algoritmii 1P*, *1Pi* şi *1Ps*.

Şi pentru *algoritmul 2D* s-a introdus pasul suplimentar la calcularea coeficienţilor (denumit *algoritmul 2D cu pas suplimentar*). Semnalele sinus şi cosinus cu M=12 eşantioane/perioadă au fost interpolate cu acesta. Se observă o reducere a valorilor erorilor de interpolare la utilizarea *algoritmului 2D cu pas suplimentar* faţă de *algoritmul 2D*. În figura 5.4 se compară erorile obţinute pentru

semnalul $y(k)=cos(2\pi k/12)$ în trei cazuri: interpolare cu *algoritmul 2D*, *algoritmul 2D cu pas suplimentar* şi *algoritmul 2P*. Pentru acest exemplu, introducerea pasului suplimentar la *2D* are ca efect obţinerea unor erori apropiate de cele obţinute la interpolarea cu *algoritmul 2P* (valori apropiate, dar de semn contrar).

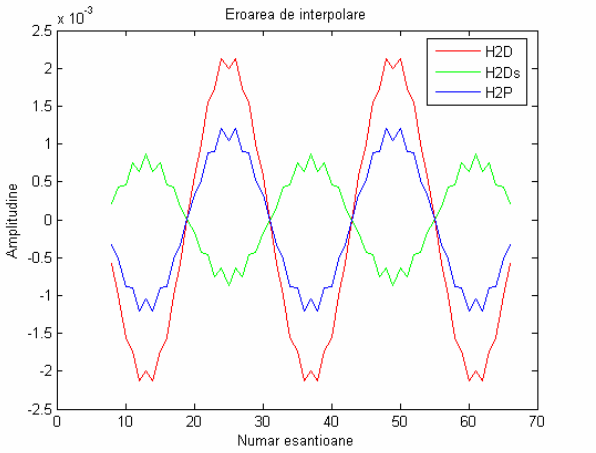


Fig.5.4. Compararea erorilor la interpolarea semnalului cosinus cu *algoritmii 2D*, *2Ds* şi *2P*.

Aplicând algoritmul 2Ds pentru semnalul $y(k)=sin(2\pi k/12)$, rezultatele sunt similare. Însă, pentru semnalul treaptă nu apar îmbunătăţiri. Din contră, sunt puncte în care erorile de interpolare sunt mai mari decât la utilizarea algoritmului iniţial.

Pentru semnalele ce au un număr mult mai mare de eşantioane pe perioadă (M=120 pentru sinus şi cosinus) nu s-a făcut interpolarea cu *algoritmul 2D cu pas suplimentar*, deoarece la interpolarea cu *algoritmul 2D* erorile sunt deja foarte mici, de ordinul 10^{-7}.

Pe baza exemplelor studiate se poate spune că aceste metode de îmbunătăţire a algoritmilor de interpolare spline cubică oferă rezultate superioare doar pentru şiruri de eşantioane provenite din semnale continue. Pentru algoritmii intercalaţi, doar în unele noduri erorile de interpolare scad foarte mult. Pentru restul punctelor, erorile scad aproximativ la jumătate. Pentru semnalele cu discontinuităţi, erorile de interpolare sunt mai mari în unele puncte din vecinătatea discontinuităţilor atunci când se calculează coeficienţii cu noile metode.

În toate cazurile prezentate nu s-au luat în discuţie erorile la capete, erori care au valori mult mai mari pentru punctele extreme ale şirului de date decât pentru restul.

5.3. Prelungirea şirului de coeficienţi

Pentru toate exemplele prezentate până acum, semnalul de la ieşire prezintă erori de interpolare mult mai mari la începutul şi sfârşitul şirului faţă de cele din rest. Acestea apar indiferent care este metoda utilizată pentru determinarea coeficienţilor. Au fost denumite *erori la capete*. S-a arătat că aceste erori sunt datorate modului în care sunt stabilite condiţiile iniţiale pentru calcularea

coeficienţilor, dar şi modului în care se obţin eşantioanele interpolate din aceşti coeficienţi. Parţial, influenţa acestor erori a fost diminuată prin modul de determinare a coeficienţilor la capetele şirului de date, mod care nu mai implică prelungirea semnalului prin oglindire.

La reconstrucţie şi interpolare, pentru determinarea fiecărui eşantion de la ieşire avem nevoie de valorile coeficienţilor în punctul respectiv, la stânga şi la dreapta acestui punct. Pentru eşantioanele de la capetele şirului, o parte din aceşti coeficienţi nu există. De obicei, acest fapt este ignorat şi valorile acestor coeficienţi sunt considerate zero. Aceasta duce la apariţia unor erori de interpolare considerabil mai mari în aceste puncte decât în restul şirului. Dacă se prelucrează un volum mare de date, aceste erori pot să nu prezinte importanţă însă, dacă se lucrează cu şiruri care au un număr mic de eşantioane, erorile la capete pot avea însemnătate.

În toate cazurile se aduc la intrarea sistemului eşantioanele $y(k)$ şi se calculează şirul de coeficienţi $c(k)$, cu k=0, 1, 2, ..., N-1. De exemplu, pentru a reconstrui semnalul de intrare, se face convoluţia între şirul coeficienţilor $c(k)$ şi $b_1^3(k)$, dat de relaţia (3.70). Fiecare eşantion al semnalului reconstituit $y_r(k)$ se calculează utilizând valorile coeficienţilor $c(k\text{-}1)$, $c(k)$ şi $c(k\text{+}1)$. Astfel, pentru primul eşantion al semnalului reconstruit $y_r(0)$ este nevoie de încă un coeficient la stânga şirului: $c(\text{-}1)$, iar pentru ultimul eşantion $y_r(N\text{-}1)$ este necesar încă un coeficient la dreapta: $c(N)$ [51].

Dacă se doreşte interpolarea semnalului cu un factor m=2, şirul de coeficienţi $c(k)$ este supraeşantionat conform relaţiei (3.48), obţinând un nou şir $c_2(k)$. Apoi se face convoluţia noului şir cu $b_2^3(k)$, dat de relaţia (3.72). Înseamnă că, pentru a calcula valorile semnalului interpolat, în fiecare punct este nevoie de 7 coeficienţi: cel din punctul curent, 3 anteriori şi 3 ulteriori. Dintre cei 3 coeficienţi: unul este calculat, iar ceilalţi 2 sunt obţinuţi prin supraeşantionare.

Astfel, pentru a reduce erorile la capete se poate prelungi şirul de coeficienţi la capete. Practic, mai sunt necesare valorile coeficienţilor $c(\textit{-1})$ şi $c(N)$.

Pentru a determina aceste valori vom porni de la relaţia (4.31) utilizată în algoritmul zero [51]. În punctul k=0 se poate scrie:

$$c(-1) = 6y(0) - 4c(0) - c(1) \tag{5.5}$$

Aceeaşi relaţie, scrisă pentru k=N-1, permite determinarea valorii coeficientului necesar la dreapta şirului:

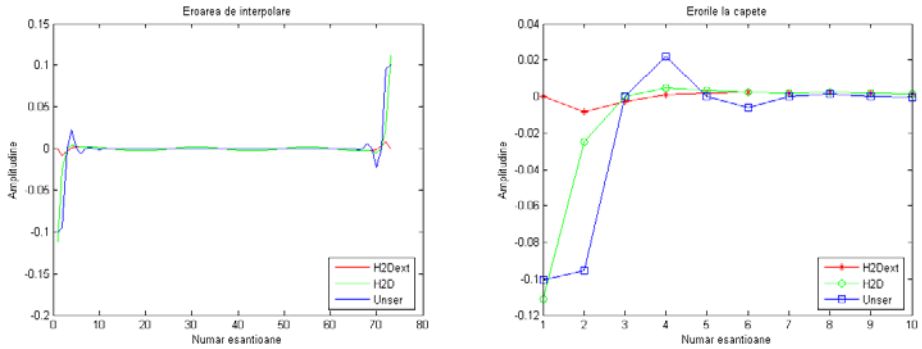
$$c(N) = 6y(N-1) - 4c(N-1) - c(N-2) \tag{5.6}$$

Pentru oricare dintre algoritmii descrişi în capitolul anterior se pot aplica relaţiile (5.5) şi (5.6) pentru a extinde şirul coeficienţilor la dreapta şi la stânga şi a reduce erorile de interpolare la capete.

Se vor prezenta în continuare câteva rezultate obţinute la interpolarea cu *algoritmul 2D* pentru care s-a făcut extinderea şirului coeficienţilor la ambele capete (denumit *algoritmul 2D extins*, notat şi *2Dext*). S-a calculat semnalul interpolat cu factor m=2 pentru cazul în care la intrare se aduc eşantioanele $y(k)$=$sin(2\pi k/M)$, unde k=0, 1, …, N-1, pentru M=120 şi N=361. La prelucrarea cu algoritmul iniţial, erorile la capete au valori de ordinul 10^{-3}, iar pentru restul eşantioanelor din şir erorile sunt de aproximativ 10^{-7}. Prin extinderea şirului de coeficienţi în ambele părţi cu câte un termen, erorile la capete scad la valori de până la 10^{-5}. Tot pentru

semnalul sinusoidal, dar cu M=12 eşantioane/perioadă, aceste erori scad de la 10^{-1} până la valori de 10^{-3}, acelaşi ordin de mărime ca şi pentru restul eşantioanelor.

În figura 5.5 se compară erorile la capete obţinute la interpolarea acestui semnal folosind *algoritmul 2D*, *algoritmul 2D extins* şi *respectiv algoritmul rapid de interpolare spline cubică* (*Unser*). În cazul interpolării cu *algoritmul Unser*, mai multe eşantioane de la capete sunt afectate de erori mai mari decât erorile pentru celelalte eşantioane, comparativ cu interpolarea prin *algoritmul 2D*. Este evident că la prelucrarea cu *algoritmul 2D extins* erorile la capete scad considerabil.



a) erorile pentru tot şirul b) detaliu la stânga şirului

Fig.5.5. Erori la capete la interpolarea $sin(2\pi k/12)$ cu algoritmii *2Dext*, *2D* şi *Unser*.

În cazul interpolării semnalului $cos(2\pi k/M)$ cu *algoritmul 2D extins* rezultatele sunt similare. Pentru M=120 se poate spune că rezultatele sunt chiar mai bune deoarece erorile la capete scad la valori de 10^{-7}, valori de acelaşi ordin fiind obţinute şi pentru restul eşantioanelor interpolate.

La interpolarea cu algoritmul 2D extins a semnalelor triunghiular şi treaptă (redate în figura 4.4) erorile la capete devin neglijabile.

Extinderea şirului de coeficienţi la ambele capete duce la scăderea semnificativă a erorilor la capete. Procesul presupune introducerea unui număr mic de operaţii suplimentare deoarece necesită calcularea doar a două elemente în plus (câte un coeficient în fiecare parte).

5.4. Abordarea analogică a procesului de derivare

În capitolul 4 au fost prezentaţi noi algoritmi de interpolare spline cubică în care se utilizează şi valorile derivatelor semnalului de intrare. S-au implementat mai multe metode de derivare numerică a funcţiilor şi au fost realizate programe în *Matlab* pentru interpolarea semnalelor de intrare. Aceste programe pot fi rescrise şi integrate în programe pentru sisteme cu procesoare numerice de prelucrare a semnalelor. Pentru multe aplicaţii, semnalele sunt achiziţionate prin intermediul unor senzori şi traductoare, filtrate, eşantionate şi convertite din semnale analogice în semnale numerice. Semnalele sunt apoi prelucrate de sistemul numeric. Există aplicaţii în care se cunosc valorile funcţiei, şi valorile derivatelor sale de ordinul 1 sau 2 (de exemplu: viteza sau acceleraţia).

Pentru algoritmii prezentaţi se presupune că se cunosc doar valorile funcţiei în anumite puncte echidistante. Valorile derivatelor se determină prin metode

numerice. Însă, derivarea semnalelor se poate realiza şi prin circuite analogice, înainte de eşantionare. Semnalul poate fi derivat analogic o dată sau de două ori, după caz. Apoi se face eşantionarea ambelor semnale (iniţial şi derivat). Astfel, la intrarea sistemului de prelucrare se vor aduce valorile semnalului şi ale derivatelor sale. Având aceste valori se pot determina coeficienţii B-spline şi se poate face interpolarea semnalelor. În figura 5.6 este prezentată schema de principiu a unui circuit de derivare care utilizează amplificatoare operaţionale.

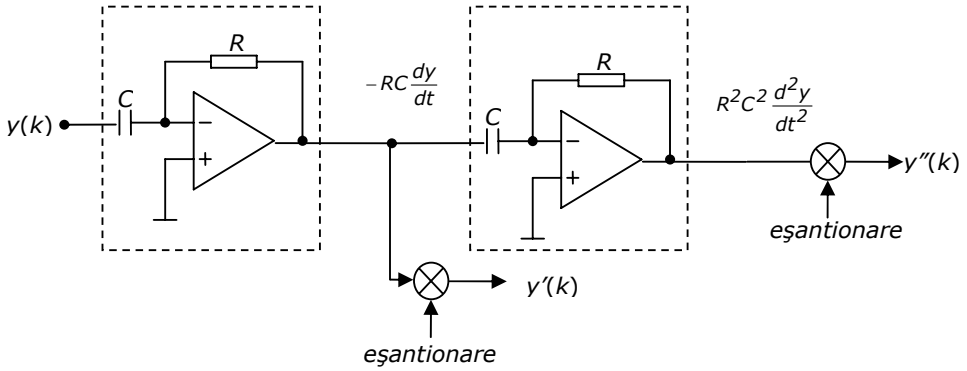


Fig.5.6. Circuit de derivare.

Procesul de derivare analogică a semnalelor prezintă şi el o serie de probleme care trebuie atent analizate.

Valorile componentelor rezistive *R* şi capacitive *C* se pot alege astfel încât să fie satisfăcută relaţia (5.7), unde T_e reprezintă perioada de eşantionare a semnalelor. În acest caz se pot calcula direct valorile coeficienţilor pentru algoritmii ce utilizează derivata a doua, deoarece în formula de calcul apare termenul *y″*(*k*)/6. (formula (4.25)).

$$R^2C^2 = \frac{1}{6}T_e^2 \tag{5.7}$$

Valorile componentelor trebuie alese cu grijă. La modificarea frecvenţei de eşantionare a sistemului de achiziţie trebuie modificate şi valorile elementelor *R* şi *C* din schema de derivare. Dacă sunt prea mici, atunci elementele parazite din circuit vor avea pondere mai mare şi semnalul va fi distorsionat. De asemenea, orice zgomot apărut la intrarea circuitului este puternic accentuat de procesul de derivare. Acest proces accentează zgomotele de înaltă frecvenţă, fapt pentru care nu se prea utilizează în practică

Eşantioanele semnalului şi eşantioanele derivatei trebuie prelevate la aceleaşi momente de timp, deoarece algoritmii folosesc valorile funcţiei şi ale derivatei sale în noduri, adică în aceleaşi puncte *k*. De aceea, amplificatoarele operaţionale folosite trebuie să fie de viteză mare pentru a evita apariţia unor defazaje semnificative între semnal şi derivata acestuia.

Dacă se folosesc semnalul şi una dintre derivate, atunci la intrarea sistemului numeric de prelucrare trebuie făcuta achiziţia a două semnale în loc de unul.

Eşantionarea derivativă este folosită în situaţii în care se cunosc semnalul şi derivata, sau derivatele sale [119, 120]. Semnalul poate fi reconstruit dacă el şi cele (n-1) derivate ale sale cunoscute au fost fiecare eşantionate la o frecvenţă de n ori mai mică decât frecvenţa cu care ar fi fost eşantionat semnalul. Astfel, pentru eşantionare şi conversie pot fi utilizate şi dispozitive care lucrează la frecvenţe de eşantionare mai reduse. Acest tip de eşantionare poate fi folosită şi în cazul funcţiilor spline, dacă avem semnalul şi derivatele sale analogice.

5.5. Algoritmi de interpolare pentru *m*>2

În toate exemplele prezentate până acum, algoritmii au fost utilizaţi pentru reconstrucţia semnalelor şi pentru interpolarea acestora cu un factor m=2. Se vor analiza în continuare câteva situaţii în care se face interpolarea semnalului cu un factor m>2. Se iau în considerare cazurile în care m este par şi respectiv, are valori de forma $m=2^r$.

Toţi algoritmii de interpolare implementaţi şi folosiţi în exemple din această lucrare urmează aceeaşi paşi (descrişi în figura 5.7):

- semnalul $y(k)$ e adus la intrarea sistemului de prefiltrare cu funcţia de transfer $H_{pf}(z)$ (specifică fiecărui algoritm) şi se obţine şirul de coeficienţi $c(k)$;
- şirul de coeficienţi este supraeşantionat cu factorul m=2;
- din noii coeficienţi se obţine semnalul interpolat prin filtrarea cu filtrul B-spline indirect $B_2^3(z)$.

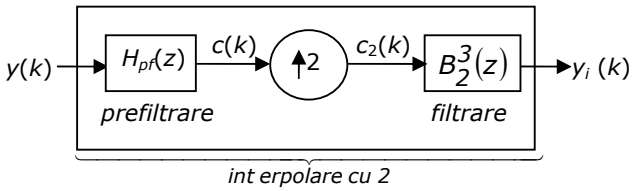


Fig.5.7. Sistem de interpolare cu m=2.

Funcţia de transfer a filtrului B-spline indirect $B_2^3(z)$ a fost prezentată în capitolul 3 (relaţia (3.71)). Pentru valori m>2, şirul de coeficienţi este supraeşantionat cu factorul m şi adus la intrarea filtrului $B_m^3(z)$. Funcţia de transfer a oricărui filtru $B_m^n(z)$ este dată de relaţia [108, 111]:

$$B_m^n(z) = \frac{1}{m^n} B_1^n(z)\left(B_m^0(z)\right)^{(n+1)} \tag{5.8}$$

unde: $$B_m^0(z)=\sum_{k=-\lfloor m/n\rfloor}^{\lfloor (m-1)/2\rfloor} z^k \qquad (5.9)$$

În cazul funcţiilor spline cubice (n=3) folosite în algoritmii de interpolare spline cubică prezentaţi, $B_1^3(z)$ este filtrul B-spline indirect cu care s-a realizat reconstrucţia semnalelor. Funcţia de transfer a acestuia a fost prezentată în capitolul 3 (relaţia (3.69)).

Pe baza formulelor (5.8) şi (5.9), pentru m=4, am determinat forma expicită a funcţiei de transfer a filtrului $B_4^3(z)$:

$$B_4^3(z)=\left[128+235\left(z+z^{-1}\right)+184\left(z^2+z^{-2}\right)+121\left(z^3+z^{-3}\right)+64\left(z^4+z^{-4}\right)+\right.$$
$$\left.+27\left(z^5+z^{-5}\right)+8\left(z^6+z^{-6}\right)+\left(z^7+z^{-7}\right)\right]\Big/384 \qquad (5.10)$$

Acesteia îi corespunde şirul $b_4^3(k)$, dat de [55]:

$$b_4^3(k)=\{1/384,\ 1/48,\ 27/384,\ 1/6,\ 121/384,\ 23/48,\ 235/384,\ 2/3,$$
$$235/384,\ 23/48,\ 121/384,\ 1/6,\ 27/384,\ 1/48,\ 1/384\} \qquad (5.11)$$

Relaţia (5.10) este complicată, iar pentru valori şi mai mari ale factorului m funcţia de transfer a filtrului B-spline indirect devine şi mai complicată. Aceasta implică mulţi termeni şi poate fi dificil de implementat.

Pentru orice valori m şi n funcţia de transfer este dată de formula (5.8), care se obţine prin înmulţirea mai multor termeni. Aceşti termeni reprezintă funcţiile de transfer ale unor filtre de ordin inferior. Una dintre proprietăţile filtrelor permite ca orice filtru de ordin superior să poată fi implementat prin legarea în cascadă a unor filtre de ordin mai mic. Funcţia de transfer a filtrului de ordin superior reprezintă produsul dintre funcţiile de transfer ale filtrelor conectate în cascadă [39, 64, 86].

Astfel, orice filtru $B_4^3(z)$ se poate implementa prin conectarea în cascadă a 4 filtre $B_4^0(z)$ şi a unui filtru $B_1^3(z)$. Pentru un semnal $y(k)$ se poate face interpolarea spline cubică cu un factor m>2 urmând paşii prezentaţi în figura 5.8. H_{pf}(z) este funcţia de transfer a sistemului care realizează operaţia de prefiltrare şi poate fi oricare dintre cele corespunzătoare algoritmilor de interpolare spline cubică prezentaţi.

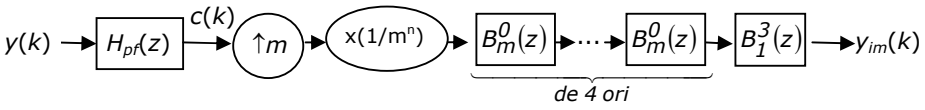


Fig. 5.8. Interpolare prin conectarea în cascadă a mai multor filtre.

Dacă factorul de interpolare are valori de forma $m=2^r$, pentru realizarea interpolării spline cubice se poate adopta şi o altă soluţie. Sistemul de interpolare cu 2 a semnalului (prezentat în figura 5.7) deja implementat poate fi folosit recursiv, de *r* ori, de fiecare dată având la intrare semnalul de la ieşirea sistemului anterior [55]. Soluţia este redată în figura 5.9.

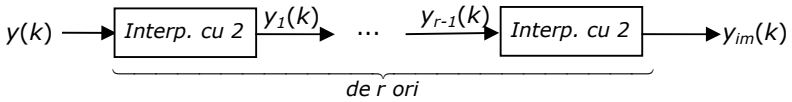


Fig. 5.9. Interpolare recursivă pentru $m=2^r$.

Ultima metodă prezentată (metoda recursivă) este eficientă din punct de vedere al implementării într-un limbaj de programare. Se scrie o secvenţă (subrutină) de program ce implementează algoritmul de interpolare cu 2. Subrutina este apoi apelată de *r* ori, de fiecare dată folosind ca variabilă de intrare şirul obţinut după rularea anterioară. Însă, această metodă presupune ca la fiecare iteraţie să se prelucreze întregul şir de date.

Cele trei variante de interpolare descrise sunt echivalente.

Metoda recursivă a fost implementată pentru interpolarea cu factorul $m=8$ a semnalului $y(k)=\cos(\alpha)$ unde $\alpha=(2\pi k/12)$ în 2 cazuri: prefiltrarea se face cu $H_{2D}(z)$, respectiv cu $H_{2P}(z)$ [55]. Astfel, *algoritmul 2D extins* (pentru factor de interpolare 2) a fost aplicat şirului de eşantioane de intrare $\{y(k)\}$, obţinând $\{y_1(k)\}$. Acest şir a fost din nou adus la intrarea sistemului de interpolare cu 2, obţinând $\{y_2(k)\}$. Acţiunea a fost repetată având la intrare $\{y_2(k)\}$. Rezultatul final îl reprezintă eşantioanele semnalului interpolat cu factorul $m=8$. S-au analizat şi comparat rezultatele fiecărei iteraţii. Erorile ce însoţesc procesul de interpolare sunt aproximativ aceleaşi, indiferent de iteraţie. În tabelul 5.2 sunt prezentate valorile semnalelor de ieşire y_i şi ale erorilor de interpolare *ein* în câteva puncte α de pe forma de undă.

Tabelul 5.2. Interpolare recursivă pentru $y(k)=cos(\alpha)$, prefiltrare cu $H_{2D}(z)$.

α	*m*	y_i	*ein*
$\pi/3$	1	0,5003	$-3{,}64\cdot10^{-4}$
	2	0,5003	$-3{,}64\cdot10^{-4}$
	4	0,5004	$-4{,}05\cdot10^{-4}$
	8	0,5004	$-4{,}04\cdot10^{-4}$
$\pi/2$	1	0	0
	2	0	0
	4	$1{,}89\cdot10^{-5}$	$-1{,}89\cdot10^{-5}$
	8	$2{,}13\cdot10^{-5}$	$-2{,}13\cdot10^{-5}$
π	1	-0,9980	$-1{,}99\cdot10^{-3}$
	2	-0,9980	$-1{,}99\cdot10^{-3}$
	4	-0,9978	$-2{,}16\cdot10^{-3}$
	8	-0,9978	$-2{,}18\cdot10^{-3}$

Sunt redate rezultatele reconstrucţiei semnalului şi ale interpolării după fiecare iteraţie. Se pot observa diferenţe nesemnificative între valorile obţinute în iteraţii diferite [55].

Pentru acelaşi şir de eşantioane de intrare s-a realizat interpolarea cu m=8 în aceeaşi manieră, folosind de această dată pentru prefiltrare sistemul cu funcţia de transfer $H_{2P}(z)$. Valorile interpolate şi erorile de interpolare pentru aceleaşi puncte α sunt redate în tabelul 5.3 [55]. Rezultatele sunt similare şi în acest caz.

Tabelul 5.3. Interpolare recursivă pentru $y(k)=cos(\alpha)$, prefiltrare cu $H_{2P}(z)$.

α	m	y_i	ein
$\pi/3$	1	0,4995	$4{,}57\cdot10^{-4}$
	2	0,4995	$4{,}57\cdot10^{-4}$
	4	0,4994	$5{,}55\cdot10^{-4}$
	8	0,4994	$5{,}67\cdot10^{-4}$
$\pi/2$	1	0	0
	2	0	0
	4	0	0
	8	0	0
π	1	-0,9989	$-1{,}04\cdot10^{-3}$
	2	-0,9989	$-1{,}04\cdot10^{-3}$
	4	-0,9988	$-1{,}14\cdot10^{-3}$
	8	-0,9988	$-1{,}15\cdot10^{-3}$

Aceeaşi metodă a fost aplicată şi pentru un semnal triunghiular. În cazul acesta apare efectul de netezire a semnalului la interpolarea cu factor de valoare mare.

Semnalele au fost prelucrate şi cu *algoritmul 2D extins* în care interpolarea cu factor m=4 se face folosind un singur filtru, respectiv filtrul B-spline indirect care are funcţia de transfer dată de relaţia (5.10). Rezultatele au fost comparate cu cele obţinute anterior. Valorile erorilor de interpolare sunt foarte apropiate. De exemplu, pentru semnalul $y(k)=\cos(2\pi k/12)$, diferenţele sunt mai mici de 10^{-3} (ordinul de mărime pentru erorile de interpolare). Şi pentru alte semnale studiate se păstrează proporţia. Aceste rezultate confirmă faptul că metodele sunt echivalente.

Pentru interpolarea cu un factor m>2 poate fi folosită oricare dintre cele trei metode prezentate. Pentru valori mari ale factorului de interpolare apar întârzieri semnificative între semnalul de ieşire şi cel de intrare. Acest aspect este relevant dacă se face implementarea algoritmului pe un sistem cu procesor numeric de semnal şi se doreşte prelucrarea în timp real a semnalului.

5.6. Concluzii

Metodele prezentate în acest capitol pot să ducă la îmbunătăţirea noilor algoritmi de interpolare spline cubică. Calcularea intercalată a coeficienţilor în cadrul algoritmilor ce folosesc valoarea derivatei întâi a funcţiei de aproximare în noduri nu oferă rezultate net mai bune pentru toţi termenii. În schimb, introducerea unui pas

suplimentar la procesul de prefiltrare are ca efect reducerea în aceeaşi proporţie a erorilor pentru toate eşantioanele semnalului interpolat. Însă, această scădere este destul de mică. În exemplele prezentate erorile scad aproximativ la jumătate. Acest lucru este valabil doar pentru semnale continue.

În cazul şirurilor de date obţinute prin eşantionarea unor semnale cu discontinuităţi, aceste metode au efect contrar, crescând valorile erorilor în unele puncte de discontinuitate şi în vecinătatea acestora.

Introducerea unui pas suplimentar înseamnă efectuarea unor operaţii în plus, ceea ce duce la timp de prelucrare mai mare, utilizarea mai multor locaţii de memorie şi adăugarea unor întârzieri pentru semnalul de ieşire. Pentru unele aplicaţii, aceste costuri suplimentare pot să nu se justifice având în vedere câştigul mic.

În schimb, pentru toate cazurile studiate, prelungirea şirului de coeficienţi cu câte un element la fiecare capăt duce la reducerea semnificativă a erorilor la capete. Acestea ajung să aibă valori apropiate de cele obţinute pentru celelalte elemente din şir.

În cazurile în care, pentru semnalul de intrare se cunosc şi derivata întâi sau a doua (de exemplu viteza sau acceleraţia), valorile acestora se pot utiliza direct pentru determinarea coeficienţilor şi interpolarea semnalului. Noii algoritmi permit calcularea digitală a derivatelor în noduri pe baza eşantioanelor semnalului. Se poate face şi derivarea analogică a semnalului, pentru ca apoi să se eşantioneze atât semnalul, cât şi derivata acestuia. Dar, această variantă are neajunsuri legate de dimensionarea schemei, alegerea componentelor, sincronizarea procesului de eşantionare şi de posibilitatea de amplificare a zgomotului.

Interpolarea cu un factor m de valoare mai mare ca 2 poate pune probleme în determinarea şi mai ales implementarea funcţiei de transfer a filtrului B-spline indirect. Am determinat forma explicită a funcţiei de transfer pentru $B_4^3(z)$, care este destul de complicată. Au fost prezentate două variante alternative la folosirea acestui filtru. Prima variantă presupune conectarea în cascadă a mai multor filtre de ordin mai mic pentru a realiza aceeaşi funcţie. A doua este mai avantajoasă din punctul de vedere al scrierii şi implementării programului dar, poate fi folosită doar pentru cazurile în care factorul de interpolare este de forma $m=2^r$. Aceasta necesită implementarea algoritmul de interpolare cu 2 şi aplicarea acestuia de r ori, de fiecare dată semnalul de ieşire fiind semnal de intrare pentru următoarea iteraţie. Variantele prezentate sunt echivalente, fapt demonstrat şi prin rezultatele experimentale obţinute.

6. IMPLEMENTAREA *ALGORITMULUI 2D* PE UN SISTEM CU PROCESOR NUMERIC DE SEMNAL (DSP)

6.1. Introducere

Procesoarele numerice de semnal (în engleză: *Digital Signal Processor* – DSP) sunt circuite integrate a căror structură internă este proiectată şi optimizată să execute într-un timp cât mai scurt operaţii specifice algoritmilor de prelucrare numerică.

Anii '70 reprezintă un început pentru tehnologia de fabricaţie şi aria de utilizare a microprocesoarelor şi a procesoarelor de semnal [125], însă primul mare succes de piaţă îl are firma *Texas Instruments* în anul 1983 cu procesorul *TMS32010* [125]. Alături de *Texas Instruments*, firme ca *Motorola, NEC, Analog Devices* continuă să îmbunătăţească performanţele procesoarelor numerice de semnal şi să permită utilizarea lor în domenii din ce în ce mai diverse.

În ultimii ani, aria de folosire a acestora s-a extins la foarte multe aplicaţii, în sectoare de activitate tot mai diverse. Se poate spune că procesoarele numerice de semnal au un caracter interdisciplinar deoarece nanotehnologia, electronica şi programarea intervin în domenii diverse precum matematica şi medicina. Sistemele care sunt actualmente pe piaţă sunt din ce în ce mai orientate pe aplicaţii. Pe lângă structura de bază, ele prezintă concepte specifice tratării unui anumit tip de probleme. Astfel, se poate vorbi de sisteme cu procesor numeric de semnal dedicate telecomunicaţiilor, procesării semnalelor audio sau prelucrării imaginilor. În continuare, se vor prezenta câteva domenii şi aplicaţii în care se folosesc microprocesoare şi procesoare numerice de semnal.

Foarte des procesoarele numerice de semnal sunt întâlnite în transferul şi prelucrarea informaţiilor din telecomunicaţii (conversaţii telefonice, semnal de televiziune, transfer de fişiere şi date diverse). Transmiterea, compresia şi prelucrarea numerică a datelor sunt avantajoase din punct de vedere al cantităţii mari de informaţii ce se vehiculează şi a costului redus al componentelor. Reprezentarea, prelucrarea şi stocarea numerică a oricărui semnal audio este importantă în primul rând pentru a preveni degradarea care este deseori asociată cu manipularea şi stocarea analogică a semnalelor. Generarea semnalului vocal şi recunoaşterea vocală sunt utilizate pentru comunicarea între om şi sisteme electronice [86].

Posibilităţile de a implementa rapid unele metode de filtrare şi de a genera uşor impulsuri de diferite forme şi frecvenţe duc la utilizarea procesoarelor în aplicaţii de eco-locaţie (radar, sonar), precum: detectarea unor obiecte, determinarea distanţei, vitezei sau a altor caracteristici ale acestora. DSP-urile au fost folosite încă de la început în domeniul reflexiei seismologice pentru detecţia unor zăcăminte terestre.

Domeniul prelucrării imaginilor a fost revoluţionat de utilizarea procesoarelor numerice de semnal, mai ales în ceea ce priveşte aplicaţiile medicale. Prelucrarea imaginilor reprezintă un subiect distinct pentru prelucrarea numerică şi utilizarea

procesoarelor numerice, deoarece acestea sunt semnale cu parametri variabili în spaţiu şi conţin o cantitate mare de informaţii. Datele obţinute prin tehnici care utilizează rezonanţa magnetică pentru investigarea corpului uman (*MRI-Magnetic Resonance Imaging*) sunt redate de obicei prin imagini. Acestea necesită o serie de prelucrări pentru care există procesoare dedicate.

Pentru imagini preluate de aparatele digitale în condiţii extreme (de exemplu: de la sateliţi, sonde spaţiale), procesoarele numerice de semnal pot îmbunătăţii calitatea acestora în ceea ce priveşte luminozitatea, contrastul, reducerea zgomotului, focalizarea [86].

În ultimele 2 decenii, volumul aplicaţiilor în care sunt utilizate procesoare de semnal a crescut foarte mult, ceea ce duce la creşterea performanţelor, mărirea vitezei de lucru, scăderea preţului de cost, reducerea consumului de putere ale acestora [26, 127]. La ora actuală, procesoarele de uz general reprezintă un competitor serios pentru procesoarele de semnal în diverse aplicaţii. În procesele de concepţie şi construcţie a acestora există un schimb de tehnici, astfel că se împrumută concepte care erau utilizate în mod exclusiv la unele sau altele pentru a face faţă cerinţelor pieţei.

6.2. Caracteristici generale ale procesoarelor numerice de semnal

Alegerea unui procesor numeric de semnal pentru realizarea unei aplicaţii în timp real este dependentă de aplicaţie şi este determinată de o serie de factori ce includ: performanţă, frecvenţă de tact, cost, consum de putere, uşurinţă de utilizare, capabilităţi de integrare şi interfaţare [84, 86]. Prelucrarea semnalelor în timp real presupune că operaţiile necesare sunt executate asupra eşantionului curent în timpul dintre 2 achiziţii consecutive. Cunoscând viteza cu care se face achiziţia semnalului de intrare, trebuie să fim siguri că secvenţa de cod care realizează operaţiile necesare obţinerii semnalului de ieşire este executată în acest interval. De aceea este foarte importantă predictibilitatea timpului de execuţie.

Viteza de prelucrare este crescută prin implementarea hardware a cât mai multor funcţiuni care se desfăşoară în paralel. Ca urmare, procesoarele de semnal conţin cât mai multe resurse integrate ce au posibilitatea să lucreze în paralel: unităţi aritmetice şi logice, multiplicator, registre interne, DMA (controler pentru accesul direct la memorie), interfeţe, circuite temporizatoare (timere), magistrale interne care permit accesul separat la zone distincte de memorie internă. Creşterea gradului de paralelism (posibilitatea de a efectua mai multe operaţii în acelaşi ciclu maşină), duce la mărirea vitezei de prelucrare, dar şi la o complexitate sporită din punct de vedere constructiv, fiind mai dificil de programat şi optimizat.

Prelucrarea numerică a semnalelor analogice (luate din sistemul pentru care se face aplicaţia) se bazează pe posibilitatea reprezentării acestor semnale prin semnale digitale (semnale eşantionate). Un semnal digital este obţinut prin eşantionarea şi cuantizarea unui semnal analogic cu ajutorul unui convertor analog numeric (CAN). Semnalele prelucrate pot fi apoi utilizate în sisteme analogice dacă se face conversia lor în semnale analogice cu ajutorul unui convertor numeric analogic (CNA). Multe traductoare şi senzori generează semnal analogic, deci împreună cu procesorul este necesar să se utilizeze cel puţin câte un convertor CAN şi unul CNA.

Principalul motiv pentru care se preferă prelucrarea semnalelor sub formă digitală este că procesarea numerică permite flexibilitate în programare. Aceeaşi structură hardware DSP poate fi folosită pentru aplicaţii diferite prin simpla schimbare a codului aflat în memorie. Prin program se pot realiza prelucrări complexe care necesită multe operaţii. Se pot implementa şi aplicaţii care nu sunt realizabile prin intermediul circuitelor analogice sau implică costuri foarte mari. Alt motiv este că acest circuit digital furnizează o ieşire mai stabilă şi tolerantă în comparaţie cu circuitele analogice.

Un alt aspect important în utilizarea procesoarelor numerice de semnal este acela al uşurinţei cu care se poate face programarea acestora. Tradiţional, programarea se făcea în limbaj de asamblare. Acest lucru presupunea o bună cunoaştere a arhitecturii interne şi a setului de instrucţiuni ale procesorului. Programarea în limbaj de asamblare permite accesul la toate resursele hard specifice, lucrându-se direct cu registre şi locaţii din memoria de date, nu cu variabile abstracte. Programele se execută mai rapid şi sunt optimizate pentru tipul de procesor pentru care au fost scrise. Dacă se doreşte mutarea aplicaţiei pe un alt tip de procesor, atunci programul trebuie rescris, ţinând cont de structura şi instrucţiunile acestuia.

Aplicaţiile ce folosesc DSP sunt din ce în ce mai complexe, ceea ce presupune scrierea a mii de linii de cod pentru a le programa. Acest lucru este greu de făcut în limbaj de asamblare şi este mai avantajoasă programarea lor într-un limbaj de nivel înalt (se utilizează limbajul *C*). Pentru aceasta s-au realizat compilatoare pentru limbaje de nivel înalt (de exemplu, compilatoare *C* pentru procesoarele *Texas Instruments* din familiile 3x, 5x şi 6x) şi medii de programare ce furnizează unelte de simulare, depanare şi urmărire a execuţiei (de exemplu, *Studio Code Composer* pentru *TMS320C5x* şi *TMS320C6x*).

Din cauza creşterii gradului de paralelism şi a arhitecturii tot mai complexe a procesoarelor moderne, este destul de greu de creat compilatoare eficiente. Pentru aplicaţii complexe care rulează în timp real, se poate face următorul compromis: scrierea programului în limbajul *C* şi utilizarea limbajului de asamblare pentru subrutinele critice.

La realizarea unei aplicaţii sunt esenţiale cunoaşterea modului de reprezentare a datelor, a caracteristicii convertoarelor, a modului în care se face transmiterea datelor între convertoare şi procesor. Din punctul de vedere al reprezentării datelor cu care lucrează, procesoarele se împart în două categorii: în virgulă fixă şi în virgulă mobilă (sau virgulă flotantă). Numerele pot fi utilizate în reprezentare binară: cod binar natural, cod binar decalat, cod complementul lui 2, cod complementul lui 1.

Unul dintre cele mai utilizate coduri este codul complementul lui 2, în care se reprezintă numere cu semn pe *N* biţi, din care primul este bit de semn. Un număr întreg cu semn poate fi scris astfel:

$$E=-2^{N-1}*b_{N-1}+2^{N-2}*b_{N-2}+\ldots\ldots+2^{1}*b_{1}+2^{0}*b_{0} \qquad (6.1)$$

iar un număr fracţionar:

$$F=-2^{0}*b_{N-1}+2^{-1}*b_{N-2}+\ldots\ldots+2^{-(N-2)}*b_{1}+2^{-(N-1)}*b_{0} \qquad (6.2)$$

Un număr mixt este un număr binar care are şi parte fracţionară şi parte întreagă, putând fi scris sub forma:

$$G=-2^{I-1}*b_{I-1}+2^{I-2}*b_{I-2}+\ldots\ldots+2^{0}*b_{0}+2^{-1}*b_{-1}+\ldots\ldots+2^{-Q}*b_{-Q} \qquad (6.3)$$

unde: I=numărul de biţi ai părţii întregi;
Q= numărul de biţi ai părţii fracţionare.

Între partea întreagă şi partea fracţionară se găseşte virgula. Pentru reprezentarea în format virgulă fixă, poziţia virgulei rămâne tot timpul aceeaşi, indiferent de rezultatul operaţiei, alocându-se un anumit număr de biţi pentru partea fracţionară şi partea întreagă. Adunarea a două numere se face după ce acestea au fost aliniate conform poziţiei virgulei. La reprezentare în format virgulă flotantă poziţia virgulei se modifică în funcţie de valoarea numărului, ele fiind reprezentate prin mantisă şi exponent. În virgulă fixă scara de reprezentare este liniară, iar cuantizarea este uniformă. În virgulă flotantă scara de reprezentare este neliniară (în progresie geometrică), iar cuantizarea este neuniformă.

Problemele care apar se datorează în principal faptului că numerele se reprezintă pe un număr finit de biţi. Intervin erori încă din procesul de eşantionare şi cuantizare la conversia analog-numerică (erori de cuantizare, erori de trunchiere), care depind de performanţele convertorului. Se pot folosi doar unele valori dintr-un anumit interval, valori care depind de numărul de biţi şi de modul de reprezentare. De exemplu: în virgulă fixă, pe 16 biţi putem reprezenta valori între 0 şi 65535 în cod binar natural sau valori între –32768 şi 32767 în cod complementul lui doi (numere cu semn). În urma efectuării unor operaţii pot să apară rezultate care depăşesc intervalele de reprezentare şi, dacă nu se ţine cont de acest aspect, pot să apară erori foarte mari în execuţia unor programe.

Acestea sunt câteva dintre problemele ce trebuiesc luate în considerare atunci când se doreşte implementarea unor algoritmi clasici pe un sistem de dezvoltare dat.

6.3. Descrierea sistemului de prelucrare numerică

Pentru algoritmii de interpolare spline cubică dezvoltaţi s-a luat în calcul posibilitatea de implementare a acestora pe un sistem cu procesor numeric de semnal. S-a utilizat sistemul *TMS320C5416 DSK* care are ca element central procesorul numeric de semnal *TMS320VC5416*.

Procesorul lucrează în virgulă fixă, având o arhitectură Harvard îmbunătăţită cu o magistrală de program, trei magistrale de date, patru magistrale de adrese şi memorii interne. Acestea permit execuţia instrucţiunilor în tehnica pipeline (operarea în paralel). Frecvenţa de tact a procesorului este de 160 MHz.

Unitatea centrală de prelucrare conţine o unitate aritmetică şi logică de 40 biţi şi un multiplicator paralel cu operanzi pe 17 biţi. Majoritatea liniilor de date şi a regiştrilor lucrează cu operanzi pe 16 de biţi, unii având posibilitatea de extindere pentru operanzi pe 40 de biţi.

Procesorul are un controler pentru acces direct la memorie DMA (în engleză: *Direct Memory Access*) care permite transferul către şi din memoria internă, perifericele interne sau componentele externe fără intervenţia unităţii centrale de prelucrare. Mai sunt incluse un circuit temporizator (timer) de câte 16 biţi care pot efectua următoarele operaţii: contorizare evenimente, contorizare timp, generare impuls, generare de întreruperi pentru unitatea centrală de prelucrare, trimitere de secvenţe de sincronizare către DMA. Comunicarea cu un calculator gazdă se poate face printr-un port paralel pe 8 biţi.

Trei componente principale ale procesorului sunt porturile seriale multicanal de mare viteză care realizează comunicaţia full-duplex şi transmiterea continuă de date prin intermediul unor registre. Acesta permite interfaţarea directă cu componente externe care realizează conversia analog-numerică şi numeric-analogică sau alte funcţii.

Pentru a realiza conversia analog-numerică a semnalelor de la intrarea sistemului şi conversia numeric-analogică a semnalelor de ieşire se utilizează circuitul *PCM3002* care este inclus pe placă. Circuitul conţine două canale de conversie pe 16 biţi: unul pentru conversie analog-numerică şi unul pentru conversie numeric-analogică (convertoare delta-sigma). Semnalele analogice de intrare şi ieşire pot fi amplificate, iar amplificarea poate fi stabilită prin program. De asemenea şi frecvenţa de eşantionare este programabilă.

Sistemul *TMS320C5416 DSK* mai conţine pe placă memorii externe, port USB pentru comunicaţia între procesor şi calculatorul gazdă, conectori pentru semnalele de intrare (*Line in*, *Mic in*), respectiv ieşire (*Line out*, *Spkr out*).

Pentru procesoarele de semnal din această familie există un set complet de unelte de dezvoltare optimizate care sunt reunite în *Code Composer Studio* (CCS). Acesta este un mediu de dezvoltare ce permite editarea programelor, generarea codului sursă, urmărirea execuţiei şi depanarea programelor. *Code Composer Studio* prezintă o interfaţă *Windows* şi are următoarele componente [129]:

- compilator C eficient;
- asamblor optimizat pentru programarea în limbaj de asamblare simplificat şi editarea de legături;
- mediu de dezvoltare integrat de management de proiect, depanare, testare şi afişare grafică;
- simulator de instrucţiuni;
- software pentru lucrul în timp real;
- sistem ce permite schimbul de date între gazdă şi ţintă în timp real (RTDX);
- sistem de analiză şi vizualizare a datelor în timp real.

Code Composer Studio integrează toate uneltele gazdă şi ţintă într-un mediu unificat care simplifică configuraţia de sistem a procesorului şi proiectarea aplicaţiilor, permiţând abordarea uşoară şi rapidă printr-o interfaţă cu utilizatorul foarte prietenoasă.

6.4. Implementarea algoritmului

Sistemul *TMS320C5416 DSK* a fost folosit pentru implementarea *algoritmului 2D de interpolare spline cubică*. S-a ales acest algoritm datorită faptului că pasul de prefiltrare se face prin intermediul unui filtru cu răspuns finit la impuls simetric. Acest filtru este stabil, uşor de implementat şi are o funcţie de transfer mai simplă decât cea a filtrului utilizat în algoritmul *2P*. De asemenea, în urma testării *algoritmului 2D* în *Matlab* pe anumite tipuri de semnale des întâlnite, s-au obţinut erori de interpolare acceptabile.

Pentru a fi implementat pe procesorul numeric de semnal, programul trebuie altfel structurat şi scris în limbajul *C*. În *Matlab*, întregul şir de eşantioane este supus mai întâi operaţiei de prefiltrare, supraeşantionat şi apoi filtrat cu filtrul B-spline indirect. Pentru a putea executa programul în timp real, fiecărui eşantion de la intrare trebuie să i se aplice toate operaţiile şi să se obţină o valoare la ieşire înainte de a se face achiziţia unui nou eşantion [56].

Trebuie să se ţină cont de faptul că procesorul lucrează cu date pe 16 biţi, iar numerele cu semn sunt reprezentate în cod complementul lui 2. De asemenea, convertorul analog-numeric furnizează eşantioanele semnalului de intrare tot în cod complementul lui 2 pe 16 biţi. Procesul de conversie este însoţit de o eroare de trunchiere de maxim:

$$e_{tr} = \frac{1}{2^{15}} \cong 3 \cdot 10^{-5} \tag{6.4}$$

Pentru şirurile de date sunt necesare linii de întârziere care sunt realizate prin deplasarea datelor.

Pentru implementarea *algoritmului 2D de interpolare spline cubică* pe sistemul cu procesor numeric de semnal se realizează un proiect în mediul de dezvoltare *Code Composer Studio*. Proiectul cuprinde fişiere *heder* în care sunt predefinite funcţii *C* standard, funcţii şi macrouri utilizate pentru tratarea problemelor specifice legate de componentele procesorului şi ale plăcii (stabilirea caracteristicilor circuitului de conversie, iniţializarea portului serial multicanal, funcţii de scriere şi citire a valorilor la portul serial, alte funcţii). Principalele fişiere ale proiectului se regăsesc în *Anexa 6*. Printre acestea, este şi un fişier de comenzi care conţine comenzi şi directive de compilare, asamblare şi link-editare.

Algoritmul este scris în limbajul *C* şi presupune parcurgerea următorilor paşi:

- citirea de la intrare a unui eşantion (de la CAN);
- calcularea coeficientului B-spline corespunzător;
- determinarea valorii eşantionului de ieşire;
- trimiterea eşantionului la ieşire (la CNA).

S-au realizat 2 programe distincte: unul pentru reconstrucţia semnalului de la intrare şi unul pentru interpolarea cu factor m=2. Citirea eşantioanelor de la convertorul analog-numeric şi trimiterea acestora la convertorul numeric-analogic se face prin intermediul unor funcţii dedicate. Semnalul de ieşire va fi întârziat faţă de cel de intrare datorită modului în care sunt calculate valorile coeficienţilor şi ale eşantioanelor semnalului interpolat.

Frecvenţa la care lucrează convertoarele analog-numeric şi numeric-analogic de pe placă este stabilită în programul principal printr-o funcţie specifică şi poate fi modificată.

În continuare vor fi prezentate câteva aspecte ale rulării programelor pe simulator, pe procesor şi câteva rezultate experimentale.

6.5. Rezultate simulare

Mediul de dezvoltare *Code Composer Studio* furnizat de producător pentru sistemul prezentat permite simularea funcţionării procesorului [129]. Acest fapt este util pentru testarea, depanarea programului şi analiza rezultatelor înainte de implementarea aplicaţiei pe procesorul de semnal de pe sistemul de dezvoltare.

Programul care descrie *algoritmul 2D* este integrat într-un proiect şi compilat. Se asamblează, se link-editează şi se generează fişierul executabil, care este încărcat în memoria sistemului şi apoi rulat. În figura 6.1 este redată interfaţa mediului de dezvoltare *Code Composer Studio*.

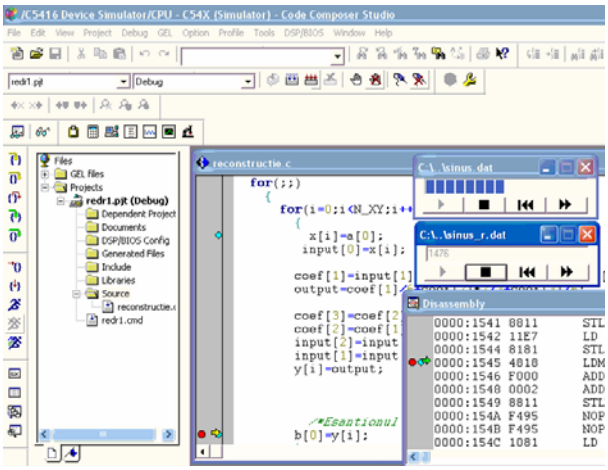


Fig. 6.1. Mediul de dezvoltare a aplicaţiilor *CCS*.

Citirea unui eşantion de la intrare şi scrierea unei valori la ieşire sunt simulate prin introducerea unor *puncte de testare* (*Toggle Probe Point*). La aceste puncte de testare sunt legate fişiere de date de intrare/ieşire. Din fişierul de intrare se transferă eşantioane într-o variabilă din program, simulând achiziţia semnalului la intrarea sistemului. În fişierul de ieşire se scriu date, simulând redarea semnalului la ieşirea sistemului. Datele (atât cele de intrare/ieşire, cât şi valorile intermediare) pot fi vizualizate prin intermediul unor grafice. Pentru ca acestea să fie actualizate la fiecare iteraţie este nevoie de introducerea în program a unui *punct de întrerupere* (*Toggle Breakpoint*). Astfel, la rularea programului se pot urmări grafic atât datele de intrare, cât şi rezultatele.

Programul de reconstrucţie a semnalului este simulat pentru cazul în care la intrare este adus un semnal sinusoidal. Graficele reprezentând semnalul de intrarea şi cel de ieşire sunt prezentate în figura 6.2 (semnalul iniţial în partea de sus şi cel de la ieşire în partea de jos).

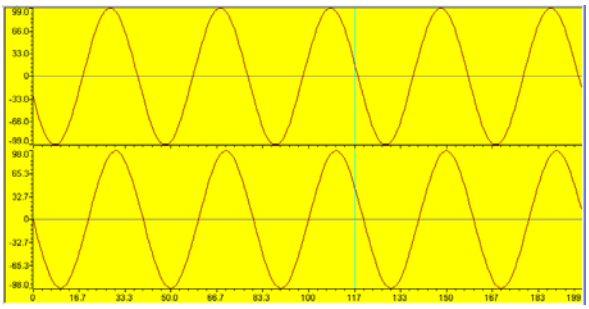


Fig. 6.2. Semnalul de intrare şi cel reconstruit (simulare).

Întârzierea între semnalul de la ieşire şi cel de la intrare este introdusă de algoritmul de interpolare şi se datorează modului în care sunt calculaţi coeficienţii şi eşantioanele semnalului de la ieşire. Şirul coeficienţilor este întârziat cu un tact faţă de semnalul de la intrare. Semnalul de la ieşire este întârziat faţă de şirul de

coeficienţi cu un număr de perioade de tact care depinde de valoarea factorului de interpolare.

Pentru interpolarea cu factor m=2, semnalele se regăsesc în figura 6.3 (semnalul de la intrare în partea de sus şi cel interpolat în partea de jos).

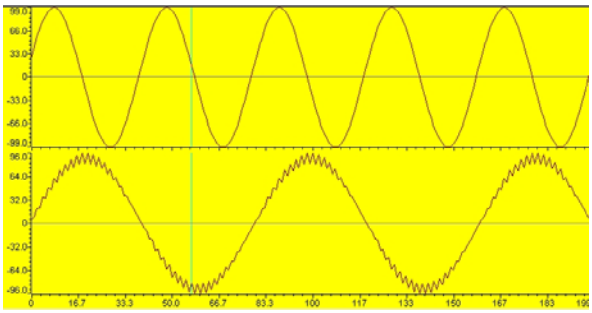


Fig. 6.3. Semnalul de intrare şi cel interpolat cu 2 (simulare).

Rezultatele sunt similare cu cele obţinute în *Matlab*. S-a trecut apoi la implementarea algoritmului pe sistemul cu procesor numeric de semnal.

6.6. Rezultate la rularea pe procesor

Pe sistemul *TMS320C5416 DSK* s-au implementat 2 proiecte: unul pentru reconstrucţia semnalului de la intrarea plăcii (fişierul sursă se regăseşte în *Anexa 6*) şi altul pentru interpolarea semnalului cu factor m=2. La intrarea *Line in* a plăcii s-a legat un generator de funcţii care furnizează semnalul de intrare. Pentru vizualizarea semnalului obţinut, la ieşirea *Line out* s-a legat un osciloscop. De la generator semnalul ajunge la circuitul *PCM3002*, la intrarea convertorului analog-numeric, este filtrat, eşantionat şi cuantizat şi apoi trimis către procesor. Semnalul prelucrat este trimis de la procesor convertorului numeric-analogic, reconstituit şi scos la ieşirea plăcii. Comunicarea între circuitul de conversie şi procesor se face prin intermediul portului serial multicanal al procesorului. Legătura între calculator şi placa cu procesor este făcută prin portul USB.

Programul de reconstrucţie a semnalului a fost compilat, asamblat şi s-a construit fişierul executabil. Acesta a fost încărcat în memoria procesorului şi rulat. La intrarea plăcii s-a adus un semnal sinusoidal cu frecvenţa f_s=100Hz şi tensiune vârf la vârf de 0,6V. Semnalele de la intrarea şi ieşirea sistemului au fost vizualizate mai întâi cu ajutorul unui osciloscop analogic. Au fost analizate 2 situaţii: frecvenţa de eşantionare f_e de 8kHz şi respectiv 48kHz. La f_e=8kHz, semnalul de la ieşire este tot un sinus cu frecvenţă 100Hz şi 0,6V tensiune vârf la vârf. Dar, peste sinus este suprapus şi un zgomot. Acest zgomot rămâne la fel şi dacă se modifică amplitudinea semnalului de la intrare, sau frecvenţa acestuia. În cazul în care semnalul de la intrare este redat la ieşire, fără să fie prelucrat, apare acelaşi zgomot suprapus peste semnal, deşi frecvenţa minimă de eşantionare pentru circuitul codificator *PCM3002* este de 4kHz.

Pentru f_e=48kHz, semnalul de la ieşirea sistemului păstrează parametrii celui de la intrare, fără să mai aibă zgomotul suprapus. La modificarea amplitudinii şi frecvenţei semnalului de la generator, semnalul de la ieşire plăcii se modifică şi el având aceleaşi caracteristici ca şi cel de la intrare. Semnalul de la ieşire este

distorsionat doar dacă semnalul de intrare depăşeşte limitele impuse de parametrii convertorului (amplitudine mai mare de 1V şi frecvenţa semnalului mai mare decât jumătate din frecvenţa de eşantionare setată).

Semnalele au fost vizualizate şi cu un osciloscop digital care permite salvarea imaginilor şi a datelor de pe ecran. Câteva imagini care au fost obţinute cu acesta sunt prezentate în figurile următoare. În fiecare figură semnalul de intrare este redat în partea de jos (canalul 1 – *C1*) şi cel prelucrat apare în partea de sus (canalul 2 – *C2*).

În figura 6.4 sunt prezentate semnalul de intrare şi semnalul de ieşire în cazul rulării programului de reconstrucţie pentru un semnal sinusoidal. La o frecvenţă de 200Hz şi o tensiune vârf la vârf de 200mV ale sinusului de la intrare, la ieşire avem aceleaşi valori pentru semnalul reconstruit.

Parametrii semnalului de intrare se păstrează la ieşire şi în cazul în care se prelucrează un sinus cu f_s=5kHz U_{vv}=0,6V (figura 6.5).

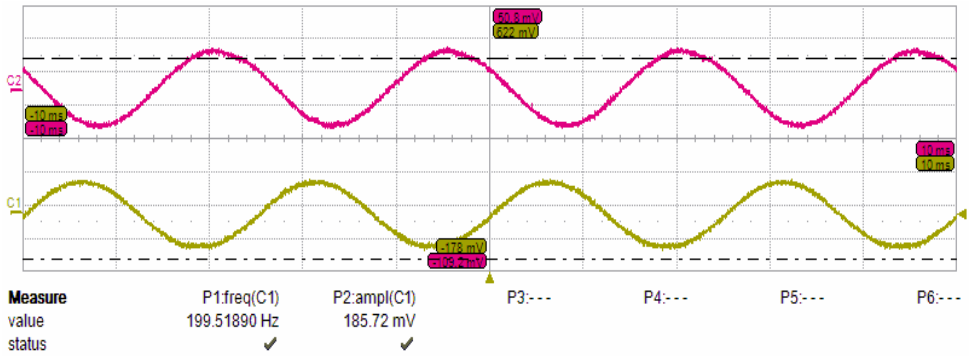


Fig. 6.4. Sinus cu f_s=200Hz.

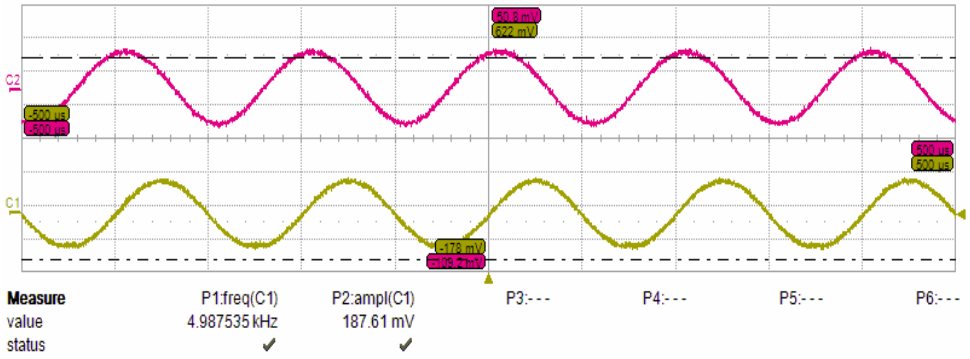


Fig. 6.5. Sinus cu f_s=5kHz.

S-a schimbat apoi forma de undă generată, aducând la intrarea sistemului un semnal triunghiular cu frecvenţa de 200Hz şi amplitudine 0,6V. Semnalul (*C1*), împreună cu rezultatul prelucrării (*C2*) sunt prezentate în figura 6.6.

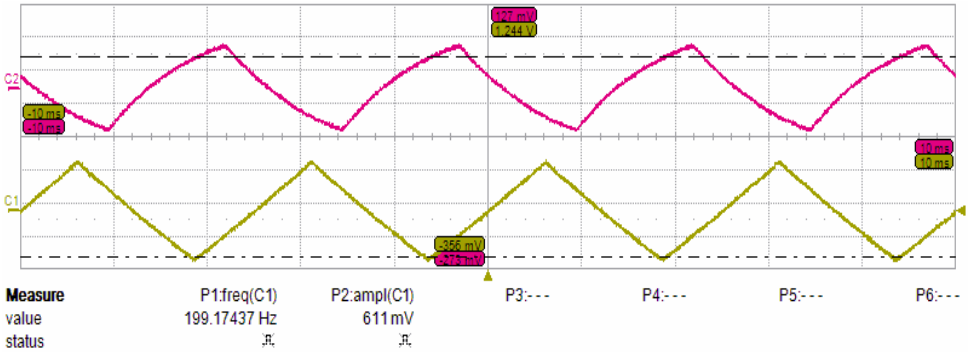


Fig. 6.6. Semnal triunghiular cu f_s=200Hz.

Se poate observa o uşoară distorsionare a semnalului triunghiular. La creşterea frecvenţei semnalului de intrare, cel de ieşire este netezit.

Pentru cazul în care la intrarea sistemului de prelucrare se aduce un semnal dreptunghiular cu frecvenţa de 1kHz, rezultatele sunt redate în figura 6.7. Se observă apariţia unor mici oscilaţii pe durata impulsurilor, oscilaţii care se atenuează rapid. În cazul creşterii frecvenţei semnalului la 5kHz, oscilaţiile sunt mai mari şi apare şi o creştere a duratei fronturilor (figura 6.8).

Şi în cazul în care semnalul de intrarea este redat la ieşire fără a fi prelucrat, pentru semnalele triunghiular şi dreptunghiular se obţin aceleaşi forme de undă ca în figurile 6.6, 6.7 şi 6.8. Aceste neteziri şi distorsionări ale semnalelor se datorează circuitului *PCM3002* care realizează conversia datelor. La intrarea circuitului, semnalul analogic este trecut printr-un filtru anti-aliere, convertit în semnal numeric, decimat şi apoi trimis la procesor. Datele primite de la procesor sunt interpolate cu un filtru digital, convertite în semnal analogic, semnal care apoi este filtrat cu un filtru trece-jos şi scos la ieşirea plăcii [127]. Toate aceste procese de filtrare duc la netezirea semnalelor testate.

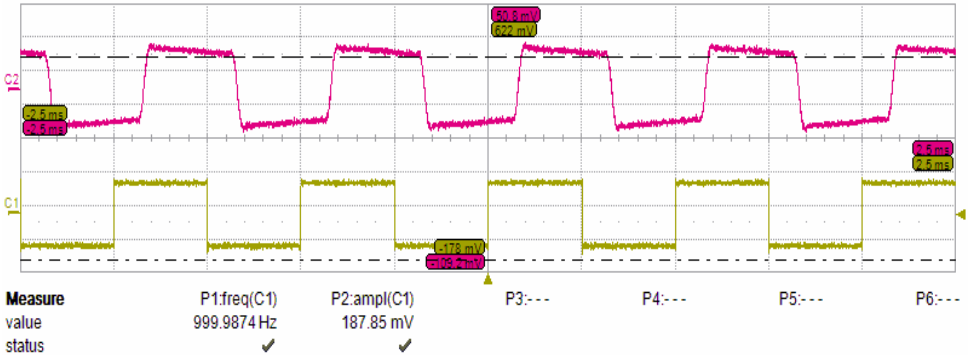


Fig. 6.7. Semnal dreptunghiular cu f_s=1kHz.

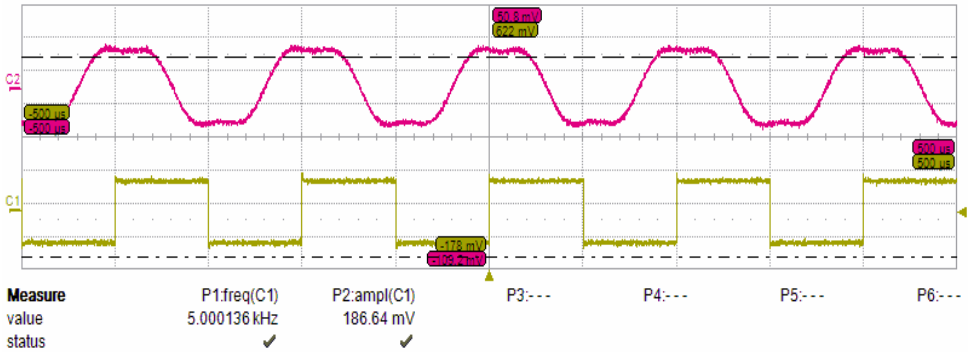


Fig. 6.8. Semnal dreptunghiular cu f_s=5kHz.

În concluzie, la prelucrarea cu *algoritmul 2D* nu sunt introduse erori suplimentare pentru semnalele de ieşire.

La analiza semnalelor vizualizate pe osciloscopul numeric trebuie să se ţină cont de o serie de factori legaţi de modul în care se face achiziţia semnalului la intrarea acestuia şi de metodele de prelucrare a datelor, de reprezentare a acestora pe ecran şi în fişierele de imagini exportate. De aceea, pentru fiecare exemplu s-au prezentat atât semnalul iniţial cât şi semnalul prelucrat cu *algoritmul 2D de interpolare spline cubică*.

6.7. Concluzii

S-a studiat posibilitatea ca unul dintre algoritmii de interpolare spline cubică dezvoltaţi în această lucrare să fie implementat pe un sistem cu procesor numeric de semnal. *Algoritmul 2D* a fost scris în limbaj *C*, integrat într-un proiect în *Code Composer Studio* şi rulat pe sistemul de dezvoltare *TMS320C5416 DSK*. S-au prezentat şi analizat câteva dintre rezultatele obţinute.

Această metodă de interpolare care presupune implementarea unor filtre FIR simetrice prezintă mai multe avantaje. Dintre acestea este de amintit faptul că algoritmul permite prelucrarea în timp real a unui semnal. Un alt avantaj este acela că algoritmul necesită realizarea unor operaţii simple şi puţine ca număr. Aceste aspecte permit o implementare facilă a algoritmului şi execuţia lui într-un timp scurt.

Rezultatele obţinute arată că mai multe tipuri de semnale (dintre cele pentru care s-a făcut testarea şi în *Matlab*) pot fi prelucrate cu acest algoritm fără apariţia unor erori suplimentare. Distorsionarea şi netezirea unor semnale se datorează operaţiilor de filtrare pe care circuitul codificator le aplică asupra semnalelor convertite, atât la intrare, cât şi la ieşire.

Proiectul poate fi optimizat astfel încât să includă şi unele unelte de analiză a datelor în timp real pentru a putea face mai multe observaţii asupra semnalelor prelucrate. De asemenea, s-ar putea estima şi timpii necesari pentru execuţia programelor. Algoritmul permite modificări şi adaptări pentru diverse situaţii şi cerinţe concrete, putând fi integrat în aplicaţii mai complexe de prelucrare numerică a semnalelor.

7. CONCLUZII ŞI CONTRIBUŢII

Domeniul prelucrării numerice a semnalelor este unul foarte vast, având numeroase aplicaţii în practică. Abordarea interpolării semnalelor prin tehnici de filtrare numerică în contextul conceptului de interpolare generalizată face ca pentru multe aplicaţii algoritmii să fie simpli, uşor de implementat şi execuţia lor să se facă într-un timp scurt. Aceste avantaje au dus la realizarea unui număr mare de noi metode de interpolare care sunt variante modificate ale unora deja cunoscute. Acestea sunt special concepute pentru anumite tipuri de aplicaţii. Un număr mare de publicaţii tratează problema interpolării spline prin diferite modele de filtre numerice, fiind prezentate şi situaţiile în care ele pot fi aplicate. Însă, acestea fac referire doar la utilizarea metodelor în domeniul prelucrării imaginilor şi sunt date exemple doar din acest domeniu.

Autoarea a ales să studieze *algoritmul rapid de interpolare spline cubică* din literatură, deoarece acesta stă la baza multor dezvoltări ulterioare. A folosit acest algoritm pentru interpolarea unor tipuri de semnale reprezentate în domeniul timp des întâlnite în teoria semnalelor. Pornind de la rezultatele obţinute, a căutat realizarea unor algoritmi similari, care să ofere soluţii simple şi rapide de interpolare pentru aplicaţii de procesare în timp real.

În urma cercetărilor şi experimentelor efectuate în acest domeniu, se pot evidenţia următoarele contribuţii ale autoarei:

- A realizat un studiu bibliografic amplu asupra metodelor şi algoritmilor de interpolare care utilizează funcţii spline, a suportului matematic şi a problemelor pe care acestea le implică. Studiul se bazează pe analiza a peste o sută de titluri bibliografice (articole, cărţi, site-uri web) şi evidenţiază actualitatea temei.
- În urma cercetărilor, experimentelor şi analizei rezultatelor, autoarea a elaborat 11 lucrări ştiinţifice (autoare şi coautoare la 10 lucrări publicate, din care una cotată ISI şi autoare a unei lucrări acceptată spre publicare la o conferinţă cotată ISI).
- Autoarea a elaborat un program în *Matlab* prin care a implementat algoritmul rapid de interpolare spline cubică preluat din literatură. În urma testării acestuia pe o serie de semnale binecunoscute, autoarea a identificat câteva neajunsuri ale metodei. Dintre acestea, o importanţă deosebită o au următoarele aspecte: algoritmul nu poate fi utilizat pentru prelucrări în timp real a semnalelor şi metoda de iniţializare introduce erori importante la capetele şirului de coeficienţi (paragraful 3.7.2) [91].
- Analizând funcţia spline cubică ce aproximează semnalul de intrare, derivatele ei de ordinul întâi şi doi şi coeficienţii B-spline ai modelului, autoarea a stabilit formule pentru iniţializarea şirului de coeficienţi care nu necesită prelungirea semnalului prin oglindire (paragraful 4.3) [91].
- Pentru metoda de prefiltrare utilizată în *algoritmul rapid de interpolare spline cubică* a propus o formulare alternativă care a dus

la elaborarea *algoritmului zero*. Sistemul este instabil, însă formula de calcul a coeficienţilor B-spline determinată în cazul acestui algoritm poate fi folosită pentru şiruri scurte de date sau ca metodă adiţională pentru îmbunătăţirea altor algoritmi (paragraful 4.4) [89].

- A stabilit formula de calcul prin care fiecare coeficient, dintr-un nod oarecare, poate fi calculat folosind valoarea semnalului şi valoarea derivatei de ordinul doi a acestuia în nodul respectiv (relaţia 4.25) [53, 90].
- Autoarea a determinat o relaţie directă între coeficienţii B-spline şi valorile în noduri ale derivatei de ordinul întâi a funcţiei de aproximare (relaţia 4.34) [90].
- Autoarea a elaborat 3 noi algoritmi de interpolare spline cubică care folosesc valorile în noduri ale derivatei de ordinul întâi, valori determinate pe baza eşantioanelor semnalului (*algoritmii 1A*, *1D*, şi *1P*). În aceste cazuri, pasul de prefiltrare a fost implementat prin filtre IIR (paragraful 4.5) [52, 90].
- A elaborat 3 noi algoritmi de interpolare spline cubică care folosesc valorile derivatei a doua în noduri, valori calculate tot pe baza eşantioanelor semnalului (*algoritmii 2A*, *2D*, şi *2P*). Pasul de prefiltrare a fost realizat prin filtre FIR, pentru *algoritmii 2D* şi *2P,* filtrele fiind simetrice. Avantajul acestora e dat de faptul că pot fi implementaţi uşor şi eficient pentru prelucrarea semnalelor în timp real (paragraful 4.6) [53, 54].
- Metodele de calcul a coeficienţilor B-spline stabilite de autoare permit determinarea locală a acestora, fapt care nu se regăseşte la nici unul din algoritmii din literatura studiată (capitolul 4).
- Autoarea a propus 2 noi variante de îmbunătăţire a metodelor de determinare a coeficienţilor în cazul noilor algoritmi (paragraful 5.2) [50].
- Autoarea a implementat o nouă metodă simplă şi eficientă de prelungire la ambele capete a şirului de coeficienţi, metodă care are ca efect reducerea erorilor la capete. Acest erori apar la implementarea tuturor algoritmilor studiaţi sau elaboraţi. Aceasta este foarte utilă, mai ales dacă prelucrarea semnalelor se face separat, pe secvenţe de eşantioane şi nu pentru tot şirul deodată (paragraful 5.3) [51].
- A determinat forma explicită a funcţiei de transfer a filtrului B-spline indirect pentru interpolare cubică cu factor de interpolare egal cu 4. Aceasta nu este redată în nici una dintre lucrările studiate (relaţia 5.10) [55].
- A propus şi comparat trei noi variante de implementare a interpolării spline cubice cu factor mai mare decât 2, de forma $m=2^r$ (paragraful 5.5) [55].
- A implementat *algoritmul 2D* original pe un sistem cu procesor numeric de semnal. Rezultatele obţinute duc la concluzia că procedeul de interpolare propus nu introduce erori suplimentare şi poate fi executat în timp real (paragraful 6.4) [56].

Interpolarea semnalelor cu algoritmii 2D şi 2P conduce la obţinerea unor erori de interpolare mai mari decât în cazul utilizării algoritmului rapid de interpolare cu funcţii spline. De cele mai multe ori, în practică, aceste erori au valori acceptabile. Avantajul major este dat de faptul că în cazul noilor algoritmi pasul de

prefiltrare se realizează prin intermediul unor filtre FIR simetrice, ceea ce le face potrivite pentru implementarea eficientă pe sisteme de prelucrare numerică şi permit realizarea de prelucrări în timp real.

Ca perspective pentru continuarea cercetării, autoarea îşi propune să optimizeze varianta prezentată de implementare pe un sistem cu procesor numeric de semnal şi să studieze rezultatele utilizării ei şi în cazul altor tipuri de semnale.

Este demonstrat că în cazul interpolării generalizate cu funcţii spline, la creşterea gradului funcţiei de interpolare se pot obţine rezultate îmbunătăţite. De aceea, autoarea intenţionează să aplice metodele dezvoltate în această teză şi în cazul funcţiilor B-spline de gradul 5. Interpolarea cu funcţii spline de gradul 5 se foloseşte în aplicaţii de prelucrare a imaginilor care se regăsesc în literatura de specialitate. Însă, pasul de prefiltrare este realizat tot cu filtre IIR, utilizând aceeaşi metodă ca în cazul interpolării spline cubice. Folosind metode similare cu cele dezvoltate în această teză, autoarea îşi propune determinarea, în viitor, a unor noi filtre FIR cu ajutorul cărora să se calculeze coeficienţii B-spline pentru interpolarea cu funcţii de gradul 5.

BIBLIOGRAFIE

[1] R. Akbar, E. Radoi, S. Azou, "Conception d'une forme d'onde IR-UWB optimisée et analyse de ses performances dans le canal IEEE 802.15.3a", *MajecSTIC 2010*Bordeaux, France, 13-15 octomber 2010.

[2] A. Aldroubi, M. Eden, M. Unser, "Discrete Spline Filters for Multiresolution and Wavelets of l_2", *SIAM Journal on Mathematical Analysis*, vol. 25, no. 5, pp. 1412-1432, September 1994.

[3] A. Aldroubi, M. Unser, M. Eden, "Cardinal Spline Filters: Stability and Convergence to the Ideal Sinc Interpolator", *Signal Processing*, vol. 28, no. 2, pp. 127-138, August 1992.

[4] Y. Barbotin, D. Van De Ville, T. Blu, M. Unser, "Fast Computation of Polyharmonic B-Spline Autocorrelation Filters", *IEEE Signal Processing Letters*, vol. 15, pp. 773-776, 2008.

[5] J.-C. Baritaux, S.C. Sekhar, M. Unser, "A Spline-Based Forward Model for Optical Diffuse Tomography", *Proceedings of the Fifth IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI'08),* Paris, France, May 14-17, 2008, pp. 384-387.

[6] O. Bernard, D. Friboulet, P. Thévenaz, M. Unser, "Variational B-Spline Level-Set: A Linear Filtering Approach for Fast Deformable Model Evolution", *IEEE Transactions on Image Processing*, vol. 18, no. 6, pp. 1179-1191, June 2009.

[7] T. Blu, P.L. Dragotti, M. Vetterli, P. Marziliano, L. Coulot, "Sparse Sampling of Signal Innovations", *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 31-40, March 2008.

[8] T. Blu, P. Thévenaz, M. Unser, "Minimum Support Interpolators with Optimum Approximation Properties", *Proceedings of the 1998 IEEE International Conference on Image Processing (ICIP'98)*, Chicago IL, USA, vol. III, pp. 242-245, October 4-7, 1998.

[9] T. Blu, P. Thévenaz, M. Unser, "Generalized Interpolation: Higher Quality at no Additional Cost", *Proceedings of the 1999 IEEE International Conference on Image Processing (ICIP'99)*, Kobe, Japan, October 25-28, 1999, vol. III, pp. 667-671.

[10] T. Blu, P. Thévenaz, M. Unser, "MOMS: Maximal-Order Interpolation of Minimal Support", *IEEE Transactions on Image Processing*, vol. 10, no. 7, pp. 1069-1080, July 2001.

[11] T. Blu, P. Thévenaz, M. Unser, "Linear Interpolation Revitalized", *IEEE Transactions on Image Processing*, vol. 13, no. 5, pp. 710-719, May 2004.

[12] T. Blu, M. Unser, "Quantitative Fourier Analysis of Approximation Techniques: Part I—Interpolators and Projectors", IEEE Transactions on Signal Processing, vol. 47, no. 10, pp. 2783-2795, October 1999.

[13] V. V. Burov, V. G. Getmanov, S. E. Orlov, V. V. Petronevich, "Method of digital filtration of experimental data sequences with the use of approximation spline functions", *Optoelectronics, Instrumentation and Data Processing*, vol. 47, no. 1, pp. 29-38, 2011.

[14] L. Condat, D. Van De Ville, "New Optimized Spline Functions for Interpolation on the Hexagonal Lattice", *Proceedings of the 2008 IEEE International Conference on Image Processing (ICIP'08),* San Diego CA, USA, October 12-15, 2008, pp. 1256-1259.

[15] C. de Boor, "Splines as Linear Combination of B-Splines. A Survey", *Approximation Theory, II*, Academic Press, New-York, 1976.

[16] C. de Boor, *A Practical Guide to Splines*, Springer-Verlag, New-York, 1978.

[17] C. de Boor, "The Condition of the B-Spline Basis for Polynomial", *SIAM J. Numer. Anal.*, vol. 25, pp. 148–152, 1988.

[18] C. de Boor, "B(asic)-Spline Basics", *Fundamental Developments of Computer-Aided Geometric Modeling*, pp. 27–49 , Academic Press, 1993.

[19] C. de Boor, R. E. Lynch, "On Splines and their Minimum Properties", *Jour. Math. Mech.*, vol. 15, pp. 953–969, 1966.

[20] Gh. Dodescu, *Metode Numerice în Algebră*, Editura Tehnică, Bucureşti, 1979.

[21] P.L. Dragotti, M. Vetterli, T. Blu, "Sampling Moments and Reconstructing Signals of Finite Rate of Innovation: Shannon Meets Strang-Fix", *IEEE Transactions on Signal Processing*, vol. 55, no. 5, part 1, pp. 1741-1757, May 2007.

[22] J. Eyre, "The Digital Processor Derby", *IEEE Spectrum*, vol. 38, no. 6, pp. 62-68, June 2001.

[23] P. J. S. G. Ferreira, "Interpolation and the Discrete Papoulis-gerchberg Algoritm", *IEEE Transactions on Signal Processing*, vol. 42, no. 10, October 1994.

[24] K. M. Flornes, "Sampling and Interpolation for Bandlimited L^p-functions", *Proceedings of the 1997 Workshop on Sampling Theory and Applications, SampTA-97*, Aveiro, Portugal, pp. 291-295, June 1997.

[25] B. Forster, T. Blu, M. Unser, "Complex B-Splines", *Applied and Computational Harmonic Analysis,* vol. 20, no. 2, pp. 261-282, March 2006.

[26] G. Frantz, "Digital Signal Processor Trends", *IEEE Micro*, vol. 20, no.6, pp. 52-59, December 2000.

[27] P. Găvruţă, O. Lipovan, P. Năslău, *Metode numerice (curs)*, Centrul de Multiplicare al Institutului Politehnic "Traian Vuia" din Timişoara, 1989.

[28] A. Gotchev, J. Vesma, T. Saramäki, K. Egiazarian, "Digital Image Resampling by modified B-Spline functions", *Norsig 2000, Nordic Signal Processing Symposium*, Kolmarden, Sweden, June 13-15, 2000, pp. 259–262.

[29] A. Gotchev, J. Vesma, T. Saramäki, K. Egiazarian, "Modified B-Spline Functions for Efficient Image Interpolation", *First IEEE Balkan Conference on Signal Processing, Communications, Circuits and Systems*, June 2-3, Istanbul, Turkey, pp. 241–244.

[30] A. Gotchev, K. Egiazarian, T. Saramäki, "Image interpolation by optimized spline-based kernels", *Advances in Signal Transforms Theory and Applications*, 2007, pp. 285-335.

[31] A. Hirabayashi, M. Unser, "Consistent Sampling and Signal Recovery", *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4104-4115, August 2007.

[32] S. Horbelt, A. Muñoz Barrutia, T. Blu, M. Unser, "Spline Kernels for Continuous-Space Image Processing", *Proceedings of the Twenty-Fifth IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'00)*, Istanbul, Turkey, June 5-9, 2000, vol. IV, pp. 2191-2194.

[33] S. Horbelt, "Splines and Wavelets for Image Warping and Projection", Swiss Federal Institute of Technology Lausanne, EPFL Thesis no. 2397 (2001), 131 p., May 23, 2001.

[34] H. Hou, H. Andrews, "Cubic Splines for Image Interpolation and Digital Filtering", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-26, No. 6, pp. 22-38, Dec. 1978.

[35] Y. K. Hu, X. M. Yu, "Perfect Spline Approximation", *Journal of Approximation Theory*, vol.121, no. 2, pp. 229-243, April 2003.

[36] K. Ichige, M. Kamada, "An Approximation for Discrete B-Splines in Time Domain", *IEEE Signal Processing Letters*, vol. 4, no. 3, pp. 82-84, March 1997.

[37] K. Ichige, M. Kamada, R. Ishii, "A Simple Scheme of Decomposing and Reconstructing Continuous-Time Signals By B-Splines", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, pp. 2391-2399, Nov. 1998.

[38] M. Jacob, T. Blu, M. Unser, "Sampling of Periodic Signals: A Quantitative Error Analysis", *IEEE Transactions on Signal Processing*, vol. 50, no. 5, pp. 1153-1159, May 2002.

[39] T. Jurca, *Aplicaţiile procesoarelor de semnal în instrumentaţia de măsurare*, Editura Politehnica Timişoara, 2001.

[40] M. Kamada, K. Toraichi, R. E. Kalman, "A Smooth Signal Generator Based on Quadratic B-Spline Functions", *IEEE Trans. on Signal Processing*, vol.43, no. 5, pp.1252-1255, May 1995.

[41] J. Kybic, T. Blu, M. Unser, "Generalized Sampling: A Variational Approach", *Proceedings of the Fourth International Conference on Sampling Theory and Applications (SampTA'01),* Orlando FL, USA, May 13-17, 2001, pp. 151-154.

[42] J. Kybic, T. Blu, M. Unser, "Generalized Sampling: A Variational Approach—Part I: Theory", *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1965-1976, August 2002.

[43] J. Kybic, T. Blu, M. Unser, "Generalized Sampling: A Variational Approach—Part II: Applications", *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1977-1985, August 2002.

[44] O. Kounchev, *Multivariate Polysplines, Applications to Numerical and Wavelet Analysis*, Elsevier Ltd., 2001.

[45] D. Larionescu, *Metode Numerice*, Editura Tehnică, Bucureşti, 1989.

[46] Th. M. Lehman, C. Gonner, K. Spitzer, "Survey: Interpolation Methods in Medical Image Processing", *IEEE Transactions on Medical Imaging*, vol. 18, pp. 1049-1075, November 1999.

[47] Th. M. Lehman, C. Gonner, K. Spitzer, "Addendum: B-Spline Interpolation in Medical Image Processing", *IEEE Transactions on Medical Imaging*, vol. 20, no. 7,pp. 660-665, July 2001.

[48] P. Marziliano, M. Vetterli, T. Blu, "Sampling and Exact Reconstruction of Bandlimited Signals with Additive Shot Noise", *IEEE Transactions on Information Theory*, vol. 52, no. 5, pp. 2230-2233, May 2006.

[49] M. Matsuo, M. Kamada, H. Habuchi, "Design Of UWB Pulses by Spline Approximation", *IEEE Wireless Communications et Networking Conference*, vol. 2, pp. 764 – 769, March 2005.

[50] **L. Mâţiu-Iovan**, "Improvements to the Algorithm that Use Divided Differences to Determine the Coefficients in B-Spline Interpolation", *Proceedings of the 15-th IMEKO TC 4 International Symposium on Novelties*

in Electrical Measurements and Instrumentations, Iaşi, vol. 2, November 18-22, 2007, pp. 616-619.

[51] **L. Mâţiu-Iovan**, C. Dughir, "Decreasing the Side Errors by Extending the Coefficients String in the B-Spline Interpolation", *Proceedings of the IX-th International Simposium "Young People and Multidisciplinary Research*", Timişoara, November 15-16, 2007, pp. 99-102.

[52] **L. Mâţiu-Iovan**, C. Dughir, G. Găşpăresc, "The First Derivative Algorithm for Calculating the B-Spline Coefficients Applied on Discontinuous Signals", *Proceedings of the First International Conference "Research People and Actual Tasks on Multidisciplinary Sciences*", Lozenec, Bulgaria, vol. 2 , June 6-8, 2007, pp. 316-320.

[53] **L. Mâţiu-Iovan**, F. M. Figură-Iliasa, C. Popa, E. Zeng, "Determining the Coefficients in B-spline Interpolation by Using the Second Derivative", *Proceedings of EUROCON 2007 The International Conference on "Computer as a tool*", Warsaw, Poland, September 9-12, 2007, pp. 142-145.

[54] **L. Mâţiu-Iovan**, F. M. Figură-Iliasa, D. Vătău, "A Method to Determine the Coefficients in B-Spline Interpolation", *Proceedings of ELECO 2007 5-th International Conference on Electrical and Electronics Engineering*, Bursa, Turcia, December 5-9, 2007, pp. 160-163.

[55] **L. Mâţiu-Iovan**, F. M. Figură-Iliasa, "Algorithms of Cubic B-Spline Interpolation Extended for m>2 ", *Proceedings of 2012 8th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP),* Poznan, Poland, July 18-20, 2012, IEEE Catalog Number: CFP1274D-DVD.

[56] **L. Mâţiu-Iovan**, "Some Aspects of Implementing a Cubic Spline Interpolation Algorithm on a DSP ", *acceptată spre publicare la ISETC 2012*.

[57] S. Mckinley, M. Levine, *Cubic Splines Interpolation*, on http://online.redwoods.cc.ca.us/instruct/darnold/laproj/fall98/Skymeg

[58] E. Meijering, "A Chronology of Interpolation: From Ancient Astronomy to Modern Signal and Image Processing", *Proceedings of the IEEE*, vol. 90, no. 3, pp. 319-342, March 2002.

[59] E. Meijering, M. Unser, "A Note on Cubic Convolution Interpolation", *IEEE Transactions on Image Processing*, vol. 12, no. 4, pp. 477-479, April 2003.

[60] Gh. Micula, *Funcţii Spline şi Aplicaţii,* Editura Tehnică, Bucureşti, 1978.

[61] Gh. Micula, S. Micula, *Handbook of Splines*, Kluwer Academic Publishers, Dordrecht-Boston-London, 1999.

[62] A. Muñoz Barrutia, T. Blu, M. Unser, "Non-Uniform to Uniform Grid Conversion Using Least-Squares Splines", *Proceedings of the Tenth European Signal Processing Conference (EUSIPCO'00)*, Tampere, Finland, September 4-8, 2000, vol. IV, pp. 1997-2000.

[63] A. Muñoz Barrutia, T. Blu, M. Unser, "Least-Squares Image Resizing Using Finite Differences", *IEEE Transactions on Image Processing*, vol. 10, no. 9, pp. 1365-1378, September 2001.

[64] I. Naforniţă, A. Câmpeanu, A. Isar, *Semnale Circuite şi Sisteme*, Editura "Politehnica", Timişoara, 1995.

[65] P. Năslău, *Metode numerice*, Editura Politehnica, Timişoara, 1995.

[66] P. Năslău, R. Negrea, L. Cădariu, B. Căruntu, D. Popescu, M. Balmez, C. Dumitraşcu, *Matematici asistate de calculator: Matlab, Mathcad, Mathematica, Maple, Derive*, Editura Politehnica, Timişoara, 2005.

[67] T. Oliveira e Silva, "On the Application of an Optimal Spline Sampling Theorem to Parametric Modeling of Nonstationary Signals ", *Proceedings of*

the 1997 Workshop on Sampling Theory and Applications, SampTA-97, Aveiro, Portugal, pp. 425-430, June 1997.

[68] J.-C. Olivo-Marin, M. Unser, L. Blanc-Féraud, A. Laine, B. Lelieveldt, "Trends in Bioimaging and Signal Processing", *IEEE Signal Processing Magazine*, vol. 28, no. 6, pp. 200 (191), November 2011.

[69] E. Pop, I. Naforniţă, V. Tiponuţ, A. Mihăescu, L. Toma, *Metode în prelucrarea numerică a semnalelor*, Editura Facla, Timişoara, 1989.

[70] R. A. Quinnell, *New DSP Architectures Go "Post-Harverd" for Higher Performance and Flexibility*, on http://www.techonline.com/community/related_content/20520

[71] U. Rajashekar, A.C. Bovik, D. Sage, M. Unser, L.J. Karam, R.L. Lagendijk, "Image Processing Education", *Handbook of Image & Video Processing (Second Edition)*, A. Bovik, Ed., Elsevier Academic Press, Amsterdam, The Netherlands, pp. 73-95, 2005.

[72] S. Ramani, D. Van De Ville, T. Blu, M. Unser, "Nonideal Sampling and Regularization Theory", *IEEE Transactions on Signal Processing*, vol. 56, no. 3, pp. 1055-1070, March 2008.

[73] S. Ramani, P. Thévenaz, M. Unser, "Regularized Interpolation for Noisy Data", *Proceedings of the Fourth IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI'07),* Arlington VA, USA, April 12-15, 2007, pp. 612-615.

[74] R. M. Rangayyan, *Biomedical Signal Analysis*, IEEE Press, Wiley-Interscience, USA, 2002.

[75] Th. J. Rivlin, *An Introduction to the Approximation of Functions*, Blaisdell Publishing Company, Waltham, Massachusetts, 1969.

[76] D. Ruijters, P. Thévenaz, "GPU Prefilter for Accurate Cubic B-Spline Interpolation", *The Computer Journal*, vol. 55, no. 1, pp. 15-20, January 2012.

[77] C.S. Seelamantula, M. Unser, "A Generalized Sampling Method for Finite-Rate-of-Innovation-Signal Reconstruction", *IEEE Signal Processing Letters*, vol. 15, pp. 813-816, 2008.

[78] C.S. Seelamantula, M. Unser, "A Method for Generalized Sampling and Reconstruction of Finite-Rate-of-Innovation Signals", *Proceedings of the Eighth International Workshop on Sampling Theory and Applications (SampTA'09)*, Marseilles, France, May 18-22, 2009.

[79] I. J. Schoenberg, "Contribution to the problem of approximation of equidistant data by analytic functions", *Quart. Applied math.*, vol. 4, pp. 45-99, 112-141, 1946.

[80] I. J. Schoenberg, "On Equidistant Cubic Spline Interpolation", *Bulletin of the American Mathematical Society*, vol. 77, no. 6, pp. 1039-1044, November 1971.

[81] I. J. Schoenberg, "Cardinal Interpolation and Spline Functions", *Journal of Approximation Theory*, vol. 2, pp. 167-206, 1969.

[82] I. J. Schoenberg, "A brief account of my life and work", *I. J. Schoenberg, Selected Papers*, ed. C. de Boor, Volume I, pp. 1-10, Birkhäuser, Boston, 1988.

[83] A. Scott, *Introduction to Splines*, http://cnx.org/content/m11153/2.1/

[84] N. Kehtarnavaz, B. Simsek, *C6X – Based Digital Signal Processing*, Prentice Hall, 2000.

[85] I. Singer**,** *Bases in Banach Spaces I,* Springer-Verlag Berlin-Heidelberg-New York, 1970.

[86] S. W. Smith, *Digital Signal Processing – A Practical Guide for Engineers and Scientists,* Newnes –An imprint of Elsevier Science, USA, 2003, http://www.dspguide.com.
[87] O. Stănăşilă, *Analiză Matematică*, Editura Didactică şi Pedagogică, Bucureşti, 1981.
[88] **L. Stoica**, "Splines Functions in Digital Signal Processing", *Proceedings of the V-th International Symposium "Young People and Multidisciplinary Research"*, pp. 164-169, Timişoara, Noiembrie 2003.
[89] **L. Stoica**, "The Study of a Divergent Algorithm for Calculating the B-Spline Coefficients", *Proceedings of the VIII-th International Symposium "Young People and Multidisciplinary Research"*, pp. 251-256, Timişoara, 11-12 Mai 2006.
[90] **L. Stoica**, "A New Algorithm for Determining the Coefficients in B-spline Interpolation", *Scientific Bulletin of the "Politehnica" University of Timişoara, Transactions on Electronics and Communications*, Vol. 51(65), No 2, "Etc2006", pp. 5-8, 2006.
[91] **L. Stoica**, "Contributions in Recursive Filtering for B-spline Interpolation in Signal Processing", *Scientific Bulletin of the "Politehnica" University of Timisoara, Transactions on Electronics and Communications*, Vol. 51(65), No 2, "Etc2006", pp. 44-49, 2006.
[92] P. Stoica, R. L. Moses**,** *Introduction to Spectral Analysis,* Prentice Hall, Upper Saddle River, New Jersey, 1997.
[93] I. Gh. Şabac, *Matematici Speciale*, vol. 2, Editura Didactica si Pedagogica Publishing House, Bucuresti, 1965
[94] I. Gh. Şabac, P. Cocârlan, O. Stănăşilă, A. Topală, *Matematici speciale*, vol. 2, Editura Didactică şi Pedagogică, Bucureşti, 1983.
[95] P.D. Tafti, S. Shirani, X. Wu, "On Interpolation and Resampling of Discrete Data", *IEEE Signal Processing Letters*, vol. 13, no. 12, pp. 733-736, December 2006.
[96] P. Thévenaz, T. Blu, M. Unser, "Image Interpolation and Resampling", *Handbook of Medical Imaging, Processing and Analysis*, I.N. Bankman, Ed., Academic Press, San Diego CA, USA, pp. 393-420, 2000.
[97] P. Thévenaz, T. Blu, M. Unser, "Interpolation Revisited", *IEEE Transactions on Medical Imaging*, vol. 19, no. 7, pp. 739-758, July 2000.
[98] M. Unser, "A General Hilbert Space Framework for the Discretization of Continuous Signal Processing Operators", *Proc. of SPIE Conf. wavelet Applications in Signal and Image Processing III*, San Diego, CA, vol. 2569, no. 6, pp. 51-61, 1995.
[99] M. Unser, "An Extension of Papoulis' Sampling Theorem for Non-Bandlimited Functions", *Proceedings of the International Workshop on Sampling Theory and Applications SampTA-97*, Aveiro, Portugal, June 16-19, pp. 10-15, 1997.
[100] M. Unser, "Splines: A Perfect Fit for Signal and Image Processing", *IEEE Signal Processing Magazine*, vol. 16, no. 6, pp. 22-38, November 1999.
[101] M. Unser, "Sampling—50 Years after Shannon", *Proceedings of the IEEE*, vol. 88, no. 4, pp. 569-587, April 2000.
[102] M. Unser, "Splines and Wavelets for Medical Imaging", *Proceedings of the Fourth IEEE EMBS Berder International Summer School on Biomedical Imaging*, June 17-24, 2000, Île de Berder, France.
[103] M. Unser, "Splines: A Perfect Fit for Medical Imaging", Plenary talk, *Proceedings of the SPIE International Symposium on Medical Imaging:*

Image Processing (MI'02), San Diego CA, USA, February 24-28, 2002, vol. 4684, part I, pp. 225-236.

[104] M. Unser, "Splines and Wavelets: New Perspectives for Pattern Recognition," *Pattern Recognition, Lecture Notes in Computer Science*, vol. 2781, pp. 244-248, Springer-Verlag Berlin Heidelberg, 2003.

[105] M. Unser, "Cardinal Exponential Splines: Part II—Think Analog, Act Digital", *IEEE Transactions on Signal Processing*, vol. 53, no. 4, pp. 1439-1449, April 2005.

[106] M. Unser, "Sampling: 60 Years After Shannon", *Plenary talk, Sixteenth International Conference on Digital Signal Processing (DSP'09*), Santorini, Greece, July 5-7, 2009.

[107] M. Unser, A. Aldroubi, "Polynomial Spline Signal Processing Algorithms", *Proceedings of the Seventeenth IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'92*), San Francisco CA, USA, March 23-26, 1992, vol. III, pp. 177-180.

[108] M. Unser, A. Aldroubi, M. Eden, "Fast B-Spline Transforms for Continuous Image Representation and Interpolation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 3, pp. 277-285, March 1991.

[109] M. Unser, A. Aldroubi, M. Eden, "Recursive Regularization Filters: Design, Properties, and Applications", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 3, pp. 272-277, March 1991.

[110] M. Unser, A. Aldroubi, M. Eden, "Polynomial Spline Signal Approximations: Filter Design and Asymptotic Equivalence with Shannon's Sampling Theorem", *IEEE Transactions on Information Theory*, vol. 38, no. 1, pp. 95-103, January 1992.

[111] M. Unser, A. Aldroubi, M. Eden, "B-Spline Signal Processing: Part I—Theory", *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 821-833, February 1993.

[112] M. Unser, A. Aldroubi, M. Eden, "B-Spline Signal Processing: Part II—Efficient Design and Applications", *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834-848, February 1993.

[113] M. Unser, T. Blu, "Cardinal Exponential Splines: Part I—Theory and Filtering Algorithms", *IEEE Transactions on Signal Processing*, vol. 53, no. 4, pp. 1425-1438, April 2005.

[114] M. Unser, T. Blu, "Generalized Smoothing Splines and the Optimal Discretization of the Wiener Filter", *IEEE Transactions on Signal Processing*, vol. 53, no. 6, pp. 2146-2159, June 2005.

[115] M. Unser, T. Blu, "Self-Similarity: Part I—Splines and Operators", *IEEE Transactions on Signal Processing*, vol. 55, no. 4, pp. 1352-1363, April 2007.

[116] M. Unser, S. Horbelt, T. Blu, "Fractional Derivatives, Splines and Tomography", *Proceedings of the Tenth European Signal Processing Conference (EUSIPCO'00)*, Tampere, Finland, September 4-8, 2000, vol. IV, pp. 2017-2020.

[117] M. Unser, J. Zerubia, "Generalized Sampling without Bandlimiting Constraints", *Proceedings of the Twenty-Second IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97),* Münich, Germany, April 21-24, 1997, vol. III, pp. 2113-2116.

[118] M. Unser, J. Zerubia, "A Generalized Sampling Theory without Band-Limiting Constraints", *IEEE Transactions on Circuits and Systems—II: Analog and Digital Signal Processing*, vol. 45, no. 8, pp. 959-969, August 1998.

[119] P.P. Vaidyanathan, "Sampling Theorems for non Bandlimited Signals: Theoretical Impact and Practical Applications", *Proceedings of the 4th International Conferences on Sampling Theory and Applications*, plenary paper, May 2001, Orlando.
[120] P.P. Vaidyanathan, B. Vrcelj, "On Sampling Theorems for non Bandlimited Signals", *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP'01*, vol. 6, pp. 3897-3900, Salt Lake City, Utah, 2001.
[121] D. Van De Ville, T. Blu, M. Unser, W. Philips, I. Lemahieu, R. Van de Walle, "Hex-Splines: A Novel Spline Family for Hexagonal Lattices", *IEEE Transactions on Image Processing*, vol. 13, no. 6, pp. 758-772, June 2004.
[122] M. Vetterli, P. Marziliano, T. Blu, "Sampling Signals with Finite Rate of Innovation", *IEEE Transactions on Signal Processing*, vol. 50, no. 6, pp.1417-1428, June 2002.
[123] B. Vrcelj, P.P. Vaidyanathan, "Efficient Implementation of All-Digital Interpolation", *IEEE Transactions on Image Processing*, Vol. 10, No. 11, pp. 1639-1646, November 2001.
[124] ***, http://bigwww.epfl.ch
[125] ***, http://www.campusprogram.com/reference/en/wikipedia
[126] ***, http:// encyclopedia.thefreedictionary.com
[127] ***, http:// www.ti.com
[128] ***, http://mathworld.wolfram.com/
[129] ***, Texas Instruments: *Software Development Systems*, Code Composer Studio CD ROM
[130] ***, http://pages.cs.wisc.edu/~deboor/
[131] http://www.uspto.gov/web/patents/patog/week27/OG/html/1380-1/US08213876-20120703.html
[132] http://www.uspto.gov/web/patents/patog/week27/OG/html/1380-1/US08213876-20120703.html

ANEXA 1. IMPLEMENTAREA ALGORITMULUI RAPID DE INTERPOLARE SPLINE CUBICĂ

Anexa 1.1. Programul în *Matlab* prin care se implementează algoritmul rapid de interpolare spline cubică

```
clear all;
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
k0=8; %k0=7
z1=-2+sqrt(3);
c=zeros(1,N);
cp=zeros(1,N);
cm=zeros(1,N);
t=0:(N-1);

sum=0;          % calcularea ccoeficienţilor
for k=1:k0
  sum=sum+y(k)*z1^(k-1);
end
cp(1)=sum;
for k=2:N
  cp(k)=y(k)+(z1*cp(k-1));
end
cm(N)=(z1/((z1^2)-1))*(cp(N)+(z1*cp(N-1)));
for k=(N-1):(-1):1
  cm(k)=z1*(cm(k+1)-cp(k));
end
c=cm*6;
save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier

s=zeros(1,N);   % reconstrucţia semnalului
b13=[1/6, 2/3, 1/6];    % B-spline _3_1
s(1)=c(1)*2/3+c(2)/6;
for k=2:N-1
  sum=0;
  for i=1:3
    j=k-i+2;
    sum=sum+c(j)*b13(i);
  end
  s(k)=sum;
end
s(N)=c(N)*2/3+c(N-1)/6;
save semnal_r.dat -ascii -double s %salvarea semnalului reconstruit în fişier
```

```
m=2;            % interpolarea cu factor m=2
D=m*N-1;
in=zeros(1,D);
cc=zeros(1,D);
b23=[1/48, 1/6, 23/48, 2/3, 23/48, 1/6, 1/48];  %B-spline_3_2
for i=1:D                % supraeşantionarea şirului de coeficienţi
  if (((i+1)/m)==round((i+1)/m))
    cc(i)=c((i+1)/m);
  end
end
in(1)=cc(1)*2/3+cc(2)*23/48+cc(3)/6+cc(4)/48;
in(2)=cc(1)*23/48+cc(2)*2/3+cc(3)*23/48+cc(4)/6+cc(5)/48;
in(3)=cc(1)/6+cc(2)*23/48+cc(3)*2/3+cc(4)*23/48+cc(5)/6+cc(6)/48;
for k=4:D-3
  sum=0;
  for i=1:7
    j=k-i+2*m;
    sum=sum+cc(j)*b23(i);
  end
  in(k)=sum;
end
in(D-2)=cc(D)/6+cc(D-1)*23/48+cc(D-2)*2/3+cc(D-3)*23/48+cc(D-
4)/6+cc(D-5)/48;
in(D-1)=cc(D)*23/48+cc(D-1)*2/3+cc(D-2)*23/48+cc(D-3)/6+cc(D-4)/48;
in(D)=cc(D)*2/3+cc(D-1)*23/48+cc(D-2)/6+cc(D-3)/48;
in=load('semnal_in.dat');

ein=yin-in;              % erorile de interpolare
save semnal_ein.dat -ascii -double ein  % salvarea erorilor de interpolare
figure
plot(t,ein);      % grafic pentru erorile de interpolare
ylabel('Amplitudine')
xlabel('Numar esantioane')
title('Eroarea de interpolare')

t4=0:2:(D-1);                % interpolarea semnalului utilizând
y4=interp1(t4,y,t,'spline');      % funcţia specifica din Matlab
figure
plot(t,y4,'*-');
ylabel('Amplitudine')
xlabel('Numar esantioane')
title('Interpolare spline Matlab')
```

Anexa 1.2. Rezultate obţinute la rularea programului

1.2.1.Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=sin(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Eşantioanele semnalului de intrare:

0.0000E+00	5.0000E-01	8.6603E-01	1.0000E+00	8.6603E-01	5.0000E-01
0.0000E+00	-5.0000E-01	-8.6603E-01	-1.0000E+00	-8.6603E-01	-5.0000E-01
0.0000E+00	5.0000E-01	8.6603E-01	1.0000E+00	8.6603E-01	5.0000E-01
0.0000E+00	-5.0000E-01	-8.6603E-01	-1.0000E+00	-8.6603E-01	-5.0000E-01
0.0000E+00	5.0000E-01	8.6603E-01	1.0000E+00	8.6603E-01	5.0000E-01
0.0000E+00	-5.0000E-01	-8.6603E-01	-1.0000E+00	-8.6603E-01	-5.0000E-01
0.0000E+00					

Valorile coeficienţilor B-spline pentru k_0=6:

-3.0223E-01	6.0435E-01	8.8481E-01	1.0526E+00	9.0495E-01	5.2379E-01
-1.1185E-04	-5.2334E-01	-9.0652E-01	-1.0467E+00	-9.0651E-01	-5.2337E-01
-4.1396E-08	5.2337E-01	9.0651E-01	1.0467E+00	9.0651E-01	5.2337E-01
-2.4319E-15	-5.2337E-01	-9.0651E-01	-1.0467E+00	-9.0651E-01	-5.2337E-01
4.1389E-08	5.2337E-01	9.0651E-01	1.0467E+00	9.0652E-01	5.2334E-01
1.1183E-04	-5.2379E-01	-9.0495E-01	-1.0526E+00	-8.8481E-01	-6.0434E-01
3.0217E-01					

Valorile coeficienţilor B-spline pentru k_0=7:

-3.0214E-01	6.0433E-01	8.8482E-01	1.0526E+00	9.0495E-01	5.2379E-01
-1.1182E-04	-5.2334E-01	-9.0652E-01	-1.0467E+00	-9.0651E-01	-5.2337E-01
-4.1384E-08	5.2337E-01	9.0651E-01	1.0467E+00	9.0651E-01	5.2337E-01
1.8965E-15	-5.2337E-01	-9.0651E-01	-1.0467E+00	-9.0651E-01	-5.2337E-01
4.1389E-08	5.2337E-01	9.0651E-01	1.0467E+00	9.0652E-01	5.2334E-01
1.1183E-04	-5.2379E-01	-9.0495E-01	-1.0526E+00	-8.8481E-01	-6.0434E-01
3.0217E-01					

Valorile coeficienţilor B-spline pentru k_0=9:

-3.0217E-01	6.0434E-01	8.8481E-01	1.0526E+00	9.0495E-01	5.2379E-01
-1.1183E-04	-5.2334E-01	-9.0652E-01	-1.0467E+00	-9.0651E-01	-5.2337E-01
-4.1388E-08	5.2337E-01	9.0651E-01	1.0467E+00	9.0651E-01	5.2337E-01
4.6854E-16	-5.2337E-01	-9.0651E-01	-1.0467E+00	-9.0651E-01	-5.2337E-01
4.1389E-08	5.2337E-01	9.0651E-01	1.0467E+00	9.0652E-01	5.2334E-01
1.1183E-04	-5.2379E-01	-9.0495E-01	-1.0526E+00	-8.8481E-01	-6.0434E-01
3.0217E-01					

Erorile de interpolare cu factor m=2, k_0=7 (erori raportate):

-1.0070E-01	-9.5585E-02	5.5511E-17	2.2076E-02	2.2204E-16	-6.1574E-03
2.2204E-16	1.3933E-03	1.1102E-16	-5.7566E-04	1.1102E-16	6.0351E-05
1.6313E-17	2.3514E-05	0.0000E+00	1.5633E-04	-2.2204E-16	2.0011E-04
-2.2204E-16	2.0291E-04	-2.2204E-16	1.4795E-04	-1.1102E-16	5.4254E-05
-4.8708E-18	-5.4223E-05	1.1102E-16	-1.4810E-04	2.2204E-16	-2.0232E-04
0.0000E+00	-2.0232E-04	1.1102E-16	-1.4811E-04	0.0000E+00	-5.4211E-05
-3.4327E-17	5.4211E-05	-1.1102E-16	1.4811E-04	0.0000E+00	2.0232E-04
-2.2204E-16	2.0232E-04	-1.1102E-16	1.4810E-04	-5.5511E-17	5.4223E-05
4.1361E-18	-5.4254E-05	1.1102E-16	-1.4795E-04	2.2204E-16	-2.0291E-04
4.4409E-16	-2.0011E-04	1.1102E-16	-1.5633E-04	5.5511E-17	-2.3511E-05
-1.7007E-18	-6.0363E-05	-1.6653E-16	5.7570E-04	-2.2204E-16	-1.3935E-03
0.0000E+00	6.1580E-03	0.0000E+00	-2.2079E-02	-1.1102E-16	9.5596E-02
1.0072E-01					

1.2.2.Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=sin(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =120 şi N=361

Erorile de interpolare cu factor m=2, k_0=7 (erori raportate) - eşantioane de la începutul şi sfârşitul şirului de date:

1.0086E-02	9.5634E-03	-1.3878E-17	-2.2247E-03	-2.7756E-17	5.9611E-04
-2.7756E-17	-1.5972E-04	-5.5511E-17	4.2803E-05	-5.5511E-17	-1.1462E-05
-5.5511E-17	3.0793E-06	-5.5511E-17	-8.1585E-07	0.0000E+00	2.2905E-07
0.0000E+00	-4.9768E-08	-5.5511E-17	2.6074E-08	-1.1102E-16	6.8498E-09
-1.1102E-16	1.3061E-08	-2.2204E-16	1.2416E-08	-1.1102E-16	1.3564E-08
-2.2204E-16	1.4186E-08	-2.2204E-16	1.4900E-08	-1.1102E-16	1.5538E-08
0.0000E+00	1.6142E-08	-1.1102E-16	1.6700E-08	0.0000E+00	1.7213E-08
-1.1102E-16	1.7679E-08	-1.1102E-16	1.8096E-08	0.0000E+00	1.8463E-08
...	...	...	...	...	...
-1.5538E-08	2.2204E-16	-1.4900E-08	3.3307E-16	-1.4186E-08	1.1102E-16
-1.3564E-08	0.0000E+00	-1.2416E-08	1.1102E-16	-1.3060E-08	1.1102E-16
-6.8519E-09	0.0000E+00	-2.6066E-08	1.1102E-16	4.9740E-08	5.5511E-17
-2.2894E-07	5.5511E-17	8.1546E-07	5.5511E-17	-3.0779E-06	5.5511E-17
1.1457E-05	5.5511E-17	-4.2783E-05	0.0000E+00	1.5965E-04	2.7756E-17
-5.9582E-04	2.7756E-17	2.2236E-03	6.9389E-18	-9.5583E-03	-1.0077E-02

1.2.3.Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Eşantioanele semnalului de intrare:

1.0000E+00	8.6603E-01	5.0000E-01	0.0000E+00	-5.0000E-01	-8.6603E-01
-1.0000E+00	-8.6603E-01	-5.0000E-01	0.0000E+00	5.0000E-01	8.6603E-01
1.0000E+00	8.6603E-01	5.0000E-01	0.0000E+00	-5.0000E-01	-8.6603E-01
-1.0000E+00	-8.6603E-01	-5.0000E-01	0.0000E+00	5.0000E-01	8.6603E-01
1.0000E+00	8.6603E-01	5.0000E-01	0.0000E+00	-5.0000E-01	-8.6603E-01
-1.0000E+00	-8.6603E-01	-5.0000E-01	0.0000E+00	5.0000E-01	8.6603E-01
1.0000E+00					

Valorile coeficienţilor B-spline pentru k_0=7:

1.0468E+00	9.0650E-01	5.2337E-01	-4.2200E-07	-5.2337E-01	-9.0651E-01
-1.0467E+00	-9.0651E-01	-5.2337E-01	-1.5618E-10	5.2337E-01	9.0651E-01
1.0467E+00	9.0651E-01	5.2337E-01	-5.8121E-14	-5.2337E-01	-9.0651E-01
-1.0467E+00	-9.0651E-01	-5.2337E-01	-1.0709E-15	5.2337E-01	9.0651E-01
1.0467E+00	9.0651E-01	5.2337E-01	1.8072E-15	-5.2337E-01	-9.0651E-01
-1.0467E+00	-9.0651E-01	-5.2337E-01	-3.8598E-15	5.2337E-01	9.0651E-01
1.0467E+00					

Erorile de interpolare cu factor m=2, k_0=7 (erori raportate):

1.5107E-01	1.9080E-02	-2.2204E-16	1.4972E-04	-1.1102E-16	5.3779E-05
1.9599E-17	-5.4095E-05	5.5511E-17	-1.4814E-04	2.2204E-16	-2.0231E-04
2.2204E-16	-2.0232E-04	0.0000E+00	-1.4811E-04	1.1102E-16	-5.4211E-05
1.0592E-17	5.4211E-05	-5.5511E-17	1.4811E-04	-2.2204E-16	2.0232E-04
0.0000E+00	2.0232E-04	0.0000E+00	1.4811E-04	0.0000E+00	5.4211E-05
-1.3027E-17	-5.4211E-05	5.5511E-17	-1.4811E-04	2.2204E-16	-2.0232E-04
2.2204E-16	-2.0232E-04	0.0000E+00	-1.4811E-04	1.1102E-16	-5.4211E-05
1.5850E-18	5.4211E-05	0.0000E+00	1.4811E-04	-2.2204E-16	2.0232E-04
-2.2204E-16	2.0232E-04	-1.1102E-16	1.4811E-04	-5.5511E-17	5.4211E-05
-4.0205E-18	-5.4211E-05	0.0000E+00	-1.4811E-04	1.1102E-16	-2.0232E-04
2.2204E-16	-2.0232E-04	2.2204E-16	-1.4811E-04	0.0000E+00	-5.4211E-05
-2.1300E-17	5.4211E-05	-1.1102E-16	1.4811E-04	-2.2204E-16	1.9088E-02
1.5108E-01					

ANEXA 2. IMPLEMENTAREA ALGORITMULUI ZERO

Anexa 2.1. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru algoritmul zero

```
clear all;
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
c2=y(3)-(16*(y(2)+y(4))-(y(1)+y(5))-30*y(3))/72;
c0=c2-(y(5)-3*y(1)-10*y(2)+18*y(3)-6*y(4))/6;
c1=(6*y(2)-c0-c2)/4;
c(1)=c0;          % determinarea coeficienţilor iniţiali
c(2)=c1;
c(3)=c2;
for i=4:1:N       % calcularea coeficienţilor
  c(i)=6*y(i-1)-4*c(i-1)-c(i-2);
end
save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier
```

Anexa 2.2. Rezultate obţinute la rularea programului

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=sin(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =120 şi N=361

Erorile de reconstrucţie corespunzătoare eşantioanelor k=30-71:

-5.4504E-09	3.9060E-08	2.7602E-07	9.4020E-08	-1.2977E-06	6.7116E-06
-1.3324E-05	2.1538E-04	1.3581E-03	2.7442E-03	1.0546E-02	5.9017E-02
-2.2285E-01	7.4314E-01	2.7071E+00	8.6691E+00	3.2629E+01	5.8779E-01
5.4464E-01	5.0000E-01	4.5399E-01	4.0674E-01	1.3107E+05	3.0902E-01
2.5882E-01	2.0791E-01	1.5643E-01	1.0453E-01	2.6844E+08	1.2246E-16
-5.2336E-02	1.7180E+10	-1.5643E-01	-2.7488E+11	-5.4976E+11	-3.0902E-01
8.7961E+12	3.5184E+13	-4.5399E-01	-5.6295E+14	-2.2518E+15	-5.8779E-01

ANEXA 3. IMPLEMENTAREA *ALGORITMILOR 1A, 1D ŞI 1P*

Anexa 3.1. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 1A*

```
clear all;
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
c2=y(3)-(y(5)-2*y(4)+y(3))/6;
c0=c2-2*(y(3)-y(2));
c1=(6*y(2)-c0-c2)/4;
c(1)=c0;          % determinarea coeficienţilor iniţiali
c(2)=c1;
c(3)=c2;
for i=4:1:N
   c(i)= 2*(y(i)-y(i-2))+c(i-2);   % calcularea coeficienţilor
end
save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier
```

Anexa 3.2. Rezultate obţinute la rularea programului *1A*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

3.4295E-02	-4.9819E-02	0.0000E+00	2.9676E-02	4.4658E-02	7.5806E-02
8.9316E-02	5.2586E-02	-1.1102E-16	-3.3763E-02	-7.7350E-02	-1.6010E-01
-2.4402E-01	-2.9258E-01	-3.3333E-01	-3.9570E-01	-4.4338E-01	-4.4183E-01
-4.2265E-01	-4.1861E-01	-3.9872E-01	-3.3226E-01	-2.5598E-01	-2.0592E-01
-1.5470E-01	-7.3443E-02	-4.4409E-16	2.9676E-02	4.4658E-02	7.5806E-02
8.9316E-02	5.2586E-02	-4.4409E-16	-3.3763E-02	-7.7350E-02	-1.6010E-01
-2.4402E-01	-2.9258E-01	-3.3333E-01	-3.9570E-01	-4.4338E-01	-4.4183E-01
-4.2265E-01	-4.1861E-01	-3.9872E-01	-3.3226E-01	-2.5598E-01	-2.0592E-01
-1.5470E-01	-7.3443E-02	1.6653E-15	2.9676E-02	4.4658E-02	7.5806E-02
8.9316E-02	5.2586E-02	6.6613E-16	-3.3763E-02	-7.7350E-02	-1.6010E-01
-2.4402E-01	-2.9258E-01	-3.3333E-01	-3.9570E-01	-4.4338E-01	-4.4183E-01
-4.2265E-01	-4.1861E-01	-3.9872E-01	-3.3226E-01	-2.5598E-01	-1.8788E-01
-1.0363E-02					

Anexa 3.3. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 1D*

```
clear all;
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
c2=y(3)-(y(4)-2*y(3)+y(2))/6;
c0=c2-y(3)+y(1);
c1=(6*y(2)-c0-c2)/4;
c(1)=c0;          % determinarea coeficienţilor iniţiali
c(2)=c1;
c(3)=c2;
for i=4:1:N
  c(i)=y(i)-y(i-2)+c(i-2);          % calcularea coeficienţilor
end
save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier
```

Anexa 3.4. Rezultate obţinute la rularea programului *1D*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

```
 1.6630E-01   2.7759E-02   0.0000E+00  -2.8690E-03  -8.1730E-03  -2.2978E-02
-3.8675E-02  -4.6199E-02  -5.2831E-02  -6.6308E-02  -7.7350E-02  -7.7918E-02
-7.5160E-02  -7.7918E-02  -7.7350E-02  -6.6308E-02  -5.2831E-02  -4.6199E-02
-3.8675E-02  -2.2978E-02  -8.1730E-03  -2.8690E-03   2.2204E-16   8.7412E-03
 1.4156E-02   8.7412E-03   0.0000E+00  -2.8690E-03  -8.1730E-03  -2.2978E-02
-3.8675E-02  -4.6199E-02  -5.2831E-02  -6.6308E-02  -7.7350E-02  -7.7918E-02
-7.5160E-02  -7.7918E-02  -7.7350E-02  -6.6308E-02  -5.2831E-02  -4.6199E-02
-3.8675E-02  -2.2978E-02  -8.1730E-03  -2.8690E-03   0.0000E+00   8.7412E-03
 1.4156E-02   8.7412E-03   2.2204E-16  -2.8690E-03  -8.1730E-03  -2.2978E-02
-3.8675E-02  -4.6199E-02  -5.2831E-02  -6.6308E-02  -7.7350E-02  -7.7918E-02
-7.5160E-02  -7.7918E-02  -7.7350E-02  -6.6308E-02  -5.2831E-02  -4.6199E-02
-3.8675E-02  -2.2978E-02  -8.1730E-03  -2.8690E-03   0.0000E+00   2.7759E-02
 1.6630E-01
```

Anexa 3.5. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 1P*

```
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
```

```
c2=y(3)-(16*(y(2)+y(4))-(y(1)+y(5))-30*y(3))/72;
c0=c2-(y(5)-3*y(1)-10*y(2)+18*y(3)-6*y(4))/6;
c1=(6*y(2)-c0-c2)/4;
c(1)=c0;          % determinarea coeficienţilor iniţiali
c(2)=c1;
c(3)=c2;
for i=4:1:(N-1) % calcularea coeficienţilor
   c(i)=(y(i-3)-8*(y(i-2)-y(i))-y(i+1))/6+c(i-2);
end
c(N)=y(N)-y(N-2)+c(N-2);

save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier
```

Anexa 3.6. Rezultate obţinute la rularea programului *1P*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

```
 1.4932E-01   1.8031E-02  -1.1102E-16   5.9426E-04   2.4929E-04  -3.0225E-04
-9.9718E-04  -1.4428E-03  -1.7451E-03  -2.4306E-03  -2.7243E-03  -3.0008E-03
-2.7422E-03  -3.0008E-03  -2.7243E-03  -2.4306E-03  -1.7451E-03  -1.4428E-03
-9.9718E-04  -3.0225E-04   2.4929E-04   6.8550E-04   7.2998E-04   1.2558E-03
 1.2465E-03   1.2558E-03   7.2998E-04   6.8550E-04   2.4929E-04  -3.0225E-04
-9.9718E-04  -1.4428E-03  -1.7451E-03  -2.4306E-03  -2.7243E-03  -3.0008E-03
-2.7422E-03  -3.0008E-03  -2.7243E-03  -2.4306E-03  -1.7451E-03  -1.4428E-03
-9.9718E-04  -3.0225E-04   2.4929E-04   6.8550E-04   7.2998E-04   1.2558E-03
 1.2465E-03   1.2558E-03   7.2998E-04   6.8550E-04   2.4929E-04  -3.0225E-04
-9.9718E-04  -1.4428E-03  -1.7451E-03  -2.4306E-03  -2.7243E-03  -3.0008E-03
-2.7422E-03  -3.0008E-03  -2.7243E-03  -2.4306E-03  -1.7451E-03  -1.4428E-03
-9.9718E-04  -3.0225E-04   2.4929E-04   1.1507E-03   4.4515E-03   3.0829E-02
 1.6712E-01
```

ANEXA 4. IMPLEMENTAREA *ALGORITMILOR 2A, 2D ŞI 2P*

Anexa 4.1. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 2A*

```
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
for i=1:1:(N-2) % calcularea coeficienţilor
   c (i)=(11*y (i)-y (i+2)+2*y (i+1))/12;
end
c (N)=y(N)-y(N-2)+c (N-2);
c (N-1)=(6*y(N-1)-c (N-2)-c(N))/4;
save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier
```

Anexa 4.2. Rezultate obţinute la rularea programului *2A*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

1.7424E-01	4.6756E-02	2.8009E-02	2.6200E-02	2.2329E-02	1.7130E-02
1.0666E-02	3.4706E-03	-3.8551E-03	-1.1119E-02	-1.7343E-02	-2.2729E-02
-2.6184E-02	-2.8249E-02	-2.8009E-02	-2.6200E-02	-2.2329E-02	-1.7130E-02
-1.0666E-02	-3.4706E-03	3.8551E-03	1.1119E-02	1.7343E-02	2.2729E-02
2.6184E-02	2.8249E-02	2.8009E-02	2.6200E-02	2.2329E-02	1.7130E-02
1.0666E-02	3.4706E-03	-3.8551E-03	-1.1119E-02	-1.7343E-02	-2.2729E-02
-2.6184E-02	-2.8249E-02	-2.8009E-02	-2.6200E-02	-2.2329E-02	-1.7130E-02
-1.0666E-02	-3.4706E-03	3.8551E-03	1.1119E-02	1.7343E-02	2.2729E-02
2.6184E-02	2.8249E-02	2.8009E-02	2.6200E-02	2.2329E-02	1.7130E-02
1.0666E-02	3.4706E-03	-3.8551E-03	-1.1119E-02	-1.7343E-02	-2.2729E-02
-2.6184E-02	-2.8249E-02	-2.8009E-02	-2.6200E-02	-2.2329E-02	-1.7130E-02
-1.0666E-02	-4.0126E-03	-4.8069E-04	-1.3466E-03	-1.1102E-16	2.8538E-02
1.6805E-01					

Anexa 4.3. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 2D*

```
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
c(3)=y(3)-(y(4)-2*y(3)+y(2))/6; % determinarea coeficienţilor iniţiali
c(1)=c(3)-y(3)+y(1);
c(2)=(6*y(2)-c(1)-c(3))/4;
for i=4:1:(N-2) % calcularea coeficienţilor
   c(i)=y(i)-(y(i+1)-2*y(i)+y(i-1))/6;
end
c(N)=y(N)-y(N-2)+c(N-2);
c(N-1)=(6*y(N-1)-c(N-2)-c(N))/4;
save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier
```

Anexa 4.4. Rezultate obţinute la rularea programului *2D*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

1.6630E-01	2.7759E-02	0.0000E+00	-1.8930E-03	-3.6499E-04	4.0001E-04
-2.2034E-17	-5.7028E-04	-9.9718E-04	-1.5580E-03	-1.7272E-03	-2.1283E-03
-1.9944E-03	-2.1283E-03	-1.7272E-03	-1.5580E-03	-9.9718E-04	-5.7028E-04
7.9981E-17	5.7028E-04	9.9718E-04	1.5580E-03	1.7272E-03	2.1283E-03
1.9944E-03	2.1283E-03	1.7272E-03	1.5580E-03	9.9718E-04	5.7028E-04
-1.5181E-16	-5.7028E-04	-9.9718E-04	-1.5580E-03	-1.7272E-03	-2.1283E-03
-1.9944E-03	-2.1283E-03	-1.7272E-03	-1.5580E-03	-9.9718E-04	-5.7028E-04
-1.5107E-16	5.7028E-04	9.9718E-04	1.5580E-03	1.7272E-03	2.1283E-03
1.9944E-03	2.1283E-03	1.7272E-03	1.5580E-03	9.9718E-04	5.7028E-04
2.7354E-16	-5.7028E-04	-9.9718E-04	-1.5580E-03	-1.7272E-03	-2.1283E-03
-1.9944E-03	-2.1283E-03	-1.7272E-03	-1.5580E-03	-9.9718E-04	-5.7028E-04
-2.9886E-16	4.0001E-04	-3.6499E-04	-1.8930E-03	-1.1102E-16	2.7759E-02
1.6630E-01					

Anexa 4.5. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 2P*

```
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
c2=y(3)-(16*(y(2)+y(4))-(y(1)+y(5))-30*y(3))/72;
c0=c2-(y(5)-3*y(1)-10*y(2)+18*y(3)-6*y(4))/6;
```

```
c1=(6*y(2)-c0-c2)/4;
 c(1)=c0; % determinarea coeficienţilor iniţiali
c(2)=c1;
c(3)=c2;
 for i=4:1:(N-2)          % calcularea coeficienţilor
   c(i)=y(i)-(16*(y(i-1)+y(i+1))-(y(i-2)+y(i+2))-30*y(i))/72;
end
c(N)=y(N)-y(N-2)+c(N-2);
c(N-1)=(6*y(N-1)-c(N-2)-c(N))/4;
 save semnal_c.dat -ascii -double c       % salvarea coeficienţilor în fişier
```

Anexa 4.6. Rezultate obţinute la rularea programului *2P*

Datele citite din fişierul „semnal.dat" corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

1.4932E-01	1.8031E-02	-1.1102E-16	6.2022E-04	4.5704E-04	3.1579E-04
-2.2034E-17	-3.2377E-04	-5.2085E-04	-8.8455E-04	-9.0215E-04	-1.2083E-03
-1.0417E-03	-1.2083E-03	-9.0215E-04	-8.8455E-04	-5.2085E-04	-3.2377E-04
5.2225E-17	3.2377E-04	5.2085E-04	8.8455E-04	9.0215E-04	1.2083E-03
1.0417E-03	1.2083E-03	9.0215E-04	8.8455E-04	5.2085E-04	3.2377E-04
-1.2405E-16	-3.2377E-04	-5.2085E-04	-8.8455E-04	-9.0215E-04	-1.2083E-03
-1.0417E-03	-1.2083E-03	-9.0215E-04	-8.8455E-04	-5.2085E-04	-3.2377E-04
-1.2332E-16	3.2377E-04	5.2085E-04	8.8455E-04	9.0215E-04	1.2083E-03
1.0417E-03	1.2083E-03	9.0215E-04	8.8455E-04	5.2085E-04	3.2377E-04
2.1802E-16	-3.2377E-04	-5.2085E-04	-8.8455E-04	-9.0215E-04	-1.2083E-03
-1.0417E-03	-1.2083E-03	-9.0215E-04	-8.8455E-04	-5.2085E-04	-3.2377E-04
-2.4334E-16	1.7668E-04	-6.5584E-04	-2.0229E-03	0.0000E+00	2.7630E-02
1.6601E-01					

ANEXA 5. IMPLEMENTAREA ALGORITMILOR ÎMBUNĂTĂŢIŢI

Anexa 5.1. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 1D intercalat*

```
clear all;
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
c2=y(3)-(y(4)-2*y(3)+y(2))/6;
c0=c2-y(3)+y(1);
c1=(6*y(2)-c0-c2)/4;
c(1)=c0;          % determinarea coeficienţilor iniţiali
c(2)=c1;
c(3)=c2;
for i=4:2:N      % calcularea coeficienţilor
   c(i)=y(i)-y(i-2)+c(i-2);
   c(i-1)=(6*y(i-1)-c(i-2)-c(i))/4;
end
c(N)=y(N)-y(N-2)+c(N-2);
 save semnal_c.dat -ascii -double c      % salvarea coeficienţilor în fişier
```

Anexa 5.2. Rezultate obţinute la rularea programului *1Di*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

1.6630E-01	2.8015E-02	2.0433E-03	3.0053E-03	1.1102E-16	-1.5453E-02
-2.3424E-02	-7.9710E-03	-5.5511E-17	-2.5987E-02	-4.5352E-02	-2.2246E-02
-1.1102E-16	-2.2246E-02	-4.5352E-02	-2.5987E-02	0.0000E+00	-7.9710E-03
-2.3424E-02	-1.5453E-02	5.5511E-17	2.5630E-03	-1.4958E-03	-1.1781E-03
1.1102E-16	-1.1781E-03	-1.4958E-03	2.5630E-03	0.0000E+00	-1.5453E-02
-2.3424E-02	-7.9710E-03	0.0000E+00	-2.5987E-02	-4.5352E-02	-2.2246E-02
0.0000E+00	-2.2246E-02	-4.5352E-02	-2.5987E-02	0.0000E+00	-7.9710E-03
-2.3424E-02	-1.5453E-02	0.0000E+00	2.5630E-03	-1.4958E-03	-1.1781E-03
2.2204E-16	-1.1781E-03	-1.4958E-03	2.5630E-03	5.5511E-17	-1.5453E-02
-2.3424E-02	-7.9710E-03	0.0000E+00	-2.5987E-02	-4.5352E-02	-2.2246E-02
-1.1102E-16	-2.2246E-02	-4.5352E-02	-2.5987E-02	0.0000E+00	-7.9710E-03
-2.3424E-02	-1.5453E-02	1.1102E-16	3.2607E-03	4.0865E-03	3.3889E-02
1.7447E-01					

Anexa 5.3. Secvenţa din programul în *Matlab* prin care se determină coeficienţii pentru *algoritmul 1D cu pas suplimentar*

```
y=load('semnal.dat');   % citire semnal de intrare
N=length(y);
c2=y(3)-(y(4)-2*y(3)+y(2))/6;
c0=c2-y(3)+y(1);
c1=(6*y(2)-c0-c2)/4;
c(1)=c0;          % determinarea coeficienţilor iniţiali
c(2)=c1;
c(3)=c2;
for i=4:1:N
  c(i)=y(i)-y(i-2)+c(i-2);          % calcularea primului set de coeficienţi
end
cn(1)=c(1);
for i=2:N-1               % calcularea celui de-al doilea set de coeficienţi
  cn(i)=(6*y(i)-c(i-1)-c(i+1))/4;
end
cn(N)=c(N);
save semnal_c.dat -ascii -double cn      % salvarea coeficienţilor în fişier
```

Anexa 5.4. Rezultate obţinute la rularea programului *1Ds*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu k=0, 1, ..., N-1

unde: M =12 şi N=37

Erorile de interpolare cu factor m=2 (erori raportate):

1.6630E-01	2.8015E-02	2.0433E-03	4.2139E-03	9.6688E-03	1.2345E-02
1.5251E-02	2.2244E-02	2.9006E-02	3.0817E-02	3.1998E-02	3.5767E-02
3.8675E-02	3.5767E-02	3.1998E-02	3.0817E-02	2.9006E-02	2.2244E-02
1.5251E-02	1.2345E-02	9.6688E-03	3.7716E-03	-1.4958E-03	-1.1781E-03
1.1102E-16	-1.1781E-03	-1.4958E-03	3.7716E-03	9.6688E-03	1.2345E-02
1.5251E-02	2.2244E-02	2.9006E-02	3.0817E-02	3.1998E-02	3.5767E-02
3.8675E-02	3.5767E-02	3.1998E-02	3.0817E-02	2.9006E-02	2.2244E-02
1.5251E-02	1.2345E-02	9.6688E-03	3.7716E-03	-1.4958E-03	-1.1781E-03
2.2204E-16	-1.1781E-03	-1.4958E-03	3.7716E-03	9.6688E-03	1.2345E-02
1.5251E-02	2.2244E-02	2.9006E-02	3.0817E-02	3.1998E-02	3.5767E-02
3.8675E-02	3.5767E-02	3.1998E-02	3.0817E-02	2.9006E-02	2.2244E-02
1.5251E-02	1.2345E-02	9.6688E-03	4.2139E-03	2.0433E-03	2.8015E-02
1.6630E-01					

Anexa 5.5. Programul în *Matlab* prin care se implementează *algoritmul 2D extins*

```
y=load('semnal.dat');
N=length(y);
c2=y(3)-(y(4)-2*y(3)+y(2))/6;
c0=c2-y(3)+y(1);
c1=(6*y(2)-c0-c2)/4;
cm1=6*y(1)-4*c0-c1;
c(1)=cm1;
c(2)=c0;
c(3)=c1;
c(4)=c2;
for i=4:1:(N-1)
  c(i+1)=y(i)-(y(i+1)-2*y(i)+y(i-1))/6;
end
c(N+1)=6*y(N-1)-4*c(N)-c(N-1);
c(N+2)=6*y(N)-4*c(N+1)-c(N);
save semnal _c.dat -ascii -double c
s=zeros(1,N);  %reconstructia semnalului
b13=[1/6, 2/3, 1/6];
for i=1:N
  sum=0;
  for k=1:3
    sum=sum+c(i+k-1)*b13(k);
  end
  s(i)=sum;
end
m=2;
D=m*N-1;
Dc=D+6;
yi=zeros(1,D);
cc=zeros(1,Dc);
b23=[1/48, 1/6, 23/48, 2/3, 23/48, 1/6, 1/48];
for i=1:Dc-1
  if (((i+1)/m)==round((i+1)/m))
    cc(i+1)=c((i+1)/m);
  end
end
for i=1:D
  sum=0;
  for k=1:7
    sum=sum+cc(i+k-1)*b23(k);
  end
  yi(i)=sum;
end
figure;
save semnal_i.dat -ascii -double yi
yin= load('semnal2.dat');        %eroarea de interpolare
ein=yin-yi;
```

```
save semnal_ein.dat -ascii -double ein
figure
plot(ein);
ylabel('Amplitudine')
xlabel('Numar esantioane')
title('Eroarea de interpolare')
```

Anexa 5.6. Rezultate obţinute la rularea programului *2Dext*

Datele citite din fişierul „semnal.dat” corespund semnalului:

$y(k)=cos(2\pi k/M)$, cu $k=0, 1, ..., N-1$

unde: $M=12$ şi $N=37$

Erorile de interpolare cu factor $m=2$ (erori raportate):

```
0.0000E+00   6.9717E-03   1.1102E-16  -1.8930E-03  -3.6499E-04   4.0001E-04
-2.2034E-17  -5.7028E-04  -9.9718E-04  -1.5580E-03  -1.7272E-03  -2.1283E-03
-1.9944E-03  -2.1283E-03  -1.7272E-03  -1.5580E-03  -9.9718E-04  -5.7028E-04
 7.9981E-17   5.7028E-04   9.9718E-04   1.5580E-03   1.7272E-03   2.1283E-03
 1.9944E-03   2.1283E-03   1.7272E-03   1.5580E-03   9.9718E-04   5.7028E-04
-1.5181E-16  -5.7028E-04  -9.9718E-04  -1.5580E-03  -1.7272E-03  -2.1283E-03
-1.9944E-03  -2.1283E-03  -1.7272E-03  -1.5580E-03  -9.9718E-04  -5.7028E-04
-1.5107E-16   5.7028E-04   9.9718E-04   1.5580E-03   1.7272E-03   2.1283E-03
 1.9944E-03   2.1283E-03   1.7272E-03   1.5580E-03   9.9718E-04   5.7028E-04
 2.7354E-16  -5.7028E-04  -9.9718E-04  -1.5580E-03  -1.7272E-03  -2.1283E-03
-1.9944E-03  -2.1283E-03  -1.7272E-03  -1.5580E-03  -9.9718E-04  -5.7028E-04
-2.9886E-16   5.7028E-04   9.9718E-04   1.3421E-03   0.0000E+00  -2.2230E-03
0.0000E+00
```

ANEXA 6. IMPLEMENTAREA ALGORITMULUI 2D PE UN PROCESOR NUMERIC DE SEMNAL

Anexa 6.1. Fişierul header *ach_red_5416cfg.h*

```
/* INPUT ach_red_5416.cdb */

#define CHIP_5416 1

/*  Include Header Files  */
#include <std.h>
#include <swi.h>
#include <tsk.h>
#include <log.h>
#include <csl_mcbsp.h>

#ifdef __cplusplus
extern "C" {
#endif

extern SWI_Obj KNL_swi;
extern TSK_Obj TSK_idle;
extern TSK_Obj Program_Principal;
extern LOG_Obj LOG_system;
extern MCBSP_Config mcbspCfg0;
extern MCBSP_Handle C54XX_DMA_MCBSP_hMcbsp;
extern void CSL_cfgInit();

#ifdef __cplusplus
}
#endif /* extern "C" */
```

Anexa 6.2. Fişier de comenzi (secvenţe)

```
/* INPUT g_s_tab_5416.cdb */
/* MODULE GBL */

SECTIONS {
   .vers (COPY): {} /* version information */
}

-priority
-llnkrtdx.a54f
-ldrivers.a54f         /* device drivers support */
-lsioboth.a54f         /* supports both SIO models */
```

```
-lbios.a54f           /* DSP/BIOS support */
-lrtdx_ext.lib        /* RTDX support */
-lcsl5416x.lib
-lrts_ext.lib         /* C and C++ run-time library support */

/* MODULE MEM */
-stack 0x100
MEMORY {
  PAGE 1:  USERREGS:   origin = 0x60,        len = 0x1a
  PAGE 1:  BIOSREGS:   origin = 0x7c,        len = 0x4
  PAGE 1:  CSLREGS:    origin = 0x7a,        len = 0x2
  PAGE 0:  VECT:       origin = 0x7f80,      len = 0x80
  PAGE 1:  IDATA:      origin = 0x80,        len = 0x7000
  PAGE 0:  IPROG:      origin = 0x7080,      len = 0xf00
  PAGE 1:  DARAM47:    origin = 0x8000,      len = 0x8000
  PAGE 0:  SARAM03:    origin = 0x28000,     len = 0x8000
  PAGE 0:  SARAM47:    origin = 0x38000,     len = 0x8000
}

/* MODULE CLK */
SECTIONS {
  .clk: {
      _CLK_start = _CLK_startNOPRSC;
      CLK_F_isr = CLK_F_isr54x;
      CLK_F_gethtime = CLK_F_getshtime;
      CLK_A_TABBEG = .;
      *(.clk)
      CLK_A_TABEND = .;
      CLK_A_TABLEN = (. - CLK_A_TABBEG) / 1;
  } > IDATA PAGE 1
}
_CLK_PRD = CLK_PRD;
_CLK_COUNTSPMS = CLK_COUNTSPMS;
_CLK_REGS = CLK_REGS;
_CLK_USETIMER = CLK_USETIMER;
_CLK_TIMERNUM = CLK_TIMERNUM;
_CLK_TCR = CLK_TCR;
_CLK_TDDR = CLK_TDDR;

/* MODULE PRD */
SECTIONS {
  .prd: {
      PRD_A_TABBEG = .;
      /* no PRD objects */
      PRD_A_TABEND = .;
      PRD_A_TABLEN = (. - PRD_A_TABBEG) / 8;
  } > IDATA PAGE 1
}

/* MODULE RTDX */
_RTDX_interrupt_mask = 0x0;
```

```
.gio:    {} > IDATA PAGE 1
      .Program_Principal$stk: {
         *(.Program_Principal$stk)
      } > IDATA PAGE 1

      .stack: fill=0xbeef {
         GBL_stackbeg = .;
         *(.stack)
         GBL_stackend = ((GBL_stackbeg + 0x100 - 1) & 0xfffe) ;
      _HWI_STKBOTTOM = GBL_stackend;
         _HWI_STKTOP = GBL_stackbeg;
      } > IDATA PAGE 1

      .rtdx_data: {}  > IDATA PAGE 1
      .TSK_idle$stk: {
         *(.TSK_idle$stk)
      } > IDATA PAGE 1

      .hst: {
          HST_A_TABBEG = .;
         _HST_A_TABBEG = .;
         /* no HST objects */
         HST_A_TABEND = .;
         _HST_A_TABEND = .;
          HST_A_TABLEN = (. - _HST_A_TABBEG) / 5;
         _HST_A_TABLEN = (. - _HST_A_TABBEG) / 5;
      } > IDATA PAGE 1

      .dsm: {} > IDATA PAGE 1
      .tsk: {
         *(.tsk)
      } > IDATA PAGE 1

      .rtdx_text: {}  > IPROG PAGE 0
      .bios:.norptb:    {} > IPROG PAGE 0
      .gblinit:    {} > IPROG PAGE 0
      .cinit:    {} > IPROG PAGE 0
      .pinit:   {} > IPROG PAGE 0
      .text:    {} > SARAM03 PAGE 0
      .sysinit:    {} > SARAM03 PAGE 0
      frt:    {} > SARAM03 PAGE 0
      .trcdata:    {} > SARAM03 PAGE 0
      .bios:    {} > SARAM03 PAGE 0
      .switch:    {} > SARAM03 PAGE 0

}
```

Anexa 6.2. Programul sursă (în *C*) - *algoritmul 2D* pentru reconstrucţia semnalului - *reconstr_5416.c*

```
#INCLUDE "ACH_RED_5416CFG.H"
#INCLUDE "DSK5416.H"
#INCLUDE "DSK5416_PCM3002.H"
DSK5416_PCM3002_CONFIG SETUP = {
   0X01FF, // SET-UP REG 0 - ATENUARE CANAL CAN STÂNGA
   0X01FF, // SET-UP REG 1 - ATENUARE CANAL CAN DREAPTA
   0X0002, // SET-UP REG 2 - PROGRAMARE CIA ÎN REGIM DE CONSUM
REDUS
   0X0000  // SET-UP REG 3 - FORMAT DATE CIA
/* PROGRAMPRINCIPAL() */
VOID PROGRAMPRINCIPAL()
{
 DSK5416_PCM3002_CODECHANDLE HCODEC;
#DEFINE N 3   /* ORDINUL B-SPLINE */
INT16 OUTPUT;
 INT16 INPUT[N];
 INT16 COEF[N+1];
INT16 INTRARE;
 INT16 IESIRE;
// INITIALIZARE CODEC
 HCODEC = DSK5416_PCM3002_OPENCODEC(0, &SETUP);
 DSK5416_PCM3002_SETFREQ(HCODEC, 48000);
//PROGRAMUL DE PRELUCRARE DE SEMNAL
 WHILE(1)
  {
   WHILE (!DSK5416_PCM3002_READ16(HCODEC, &INTRARE));
         //CITIRE DE LA CAN
      INPUT[0]=INTRARE;
   COEF[1]=INPUT[1]-(INPUT[0]-INPUT[1]-INPUT[1]+INPUT[2])*0.16667;
   OUTPUT=COEF[1]*0.16667+COEF[2]*0.66667+COEF[3]*0.16667;
      COEF[3]=COEF[2];
      COEF[2]=COEF[1];
      INPUT[2]=INPUT[1];
      INPUT[1]=INPUT[0];
      IESIRE=OUTPUT;
        //PRELUCRARE SEMNAL
   WHILE (!DSK5416_PCM3002_WRITE16(HCODEC, IESIRE));    //SCRIERE
LA CNA
  }
}
VOID MAIN()
{
// INITIALIZARE DSK
DSK5416_INIT();
}
```