LIBERTAS MATEMATICA, vol XXVII (2007)

Matlab Evaluation of the $\Gamma_k^n(x)$ Coefficients for PDE Solving by Wavelet-Galerkin Approximation

Constantin I. POPOVICI and Constantin FILIPESCU

Dedicated to Academician Radu Miron on his 80th Birthday

Abstract. This paper is a part of the research for solving PDE or ODE by wavelet-Galerkin method. Such approach needs some families of coefficients which appear in the wavelet series built by discretization of a differential equation based on mathematical-mechanical model. Following some ideas (see References) we develop many algorithms & MATLAB scripts for highly accurate calculus of needed functionals. Here we show the MATLAB algorithmic evaluation of integral type $\Gamma_k^n(x) = \int_0^x \phi^{(n)}(y-k)\phi(y)dy$.

Keywords: Wavelet-Galerkin, approximation, PDE.

Mathematics Subject Classification (2000): 65T60.

1 Introduction

Following the construction [2] of an orthonormal base of wavelet functions with compact support invented by prof. I. Daubechies, in 1996 the team of prof. Chen (Taiwan National Cheng Kung University) have proposed [1] an concise algorithmic scheme with the aim of evaluating a family of 7 functionals which can appear in the wavelet-Galerkin discretization of a differential equation. Here we show the results of efforts to construct the algorithm and programs necessary for the numerical evaluation of one of these functionals :

$$G_k^n(x) = \int_0^x f^{(n)}(y-k)f(y)dy \tag{1}$$

using MATLAB development tool. Here $f^{(n)}(u)$ represents the n^{th} derivative of the function $\phi(u)$.

2 The $\Gamma_k^n(x)$ calculation

The function is the wavelet father-function or scaling function. The scaling function has fundamental support in the interval $[0, L-1]$ and is defined by the two scale relation :

$$\phi(x) = \sum_{j=0}^{L-1} p_j \phi(2x - j). \tag{2}$$

His companion, called *mother wavelet*, ψ ,is defined on the interval [1-L/2, L-2] and satisfies the equation.

$$\psi(x) = \sum_{j=2-L}^{1} (-1)^j p_{1-j} \phi(2x - j).$$

The filter coefficients p_k can be defined (after Daubechies) for $L = 6$ as follows :

$$p_0 = \frac{1 + \sqrt{10} + \sqrt{5 + 2\sqrt{10}}}{16}, \quad p_1 = \frac{5 + \sqrt{10} + 3\sqrt{5 + 2\sqrt{10}}}{16}$$

$$p_2 = \frac{10 - 2\sqrt{10} + 2\sqrt{5 + 2\sqrt{10}}}{16}, \quad p_3 = \frac{10 - 2\sqrt{10} - 2\sqrt{5 + 2\sqrt{10}}}{16}$$

$$p_4 = \frac{5 + \sqrt{10} - 3\sqrt{5 + 2\sqrt{10}}}{16}, \quad p_5 = \frac{1 + \sqrt{10} - \sqrt{5 + 2\sqrt{10}}}{16}$$

The property $\sum_n p_n = 2$ is, obviously, verified in this particular case. The evaluation of $\Gamma_k^n(x)$ use the particular values $L = 6$, $x = 1, 2, \ldots, L-1$, $n = 1$. It is quite easy to derive the next properties :

$$\Gamma_k^n(x) = \Gamma_k^n(L-1) \text{ for } x \geq L-1 \tag{3}$$

$$\Gamma_k^n(x) = 0 \text{ for } |k| \geq L-1 \text{ or } x \leq 0 \text{ or } x \leq k \tag{4}$$

$$\Gamma_{-k}^n(L-1) = (-1)^k \Gamma_k^n(L-1) \text{ for } x \geq 0 \tag{5}$$

$$\Gamma_{-k}^n(x) = (-1)^n \Gamma_k^n(L-1) \text{ for } x - k \geq L-1 \tag{6}$$

The above relations further implies:

$$\begin{aligned}
\Gamma_k^n(x) &= 2^n \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} p_i p_j \int_0^x \phi^{(n)}(2y - 2k - i)\phi(2y - j)dy = \\
&= 2^{n-1} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} p_i p_j \int_0^{2x-j} \phi^{(n)}(y - 2k - i + j)\phi(y)dy = \\
&= 2^{n-1} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} p_i p_j \Gamma_{2k+i-j}^n.
\end{aligned} \tag{7}$$

For $x = L - 1$ and $n = 1$ one obtain from (3)-(6) and (7) a linear system wich leads (after we remove it's indetermination) to a set of values for G_k^1 where $k \in [-4, 4]$. The corresponding MatLab script is the following:

```
% p5gama = GAMA coefficients calculus
clc, clear all
L=6; n=1;
load p_i;
Lm1=L-1; Lm2=L-2;
D=zeros(Lm1,Lm1);
for l=0:1:Lm2
   for m=0:1:Lm2
      s1=0;   s2=0;
      for i=0:1:Lm1
         for j=0:1:Lm1
            if 2*l+i-j == m
               s1=s1+p(i+1)*p(j+1);
            end
            if 2*l+i-j == -m
               s2=s2+p(i+1)*p(j+1);
            end
         end
      end
      D(l+1,m+1)=2^(n-1)*(s1+s2*(-1)^n);
   end
end
[V1,D1]=eig(D);
format long
Gk1Lm1=V1(:,1);
s=0;
for k=2:Lm1
   s=s+(k-1)^n*Gk1Lm1(k);
end
% 2. normalization of the eigenvector coresp. eigenvalue=1
Gk1Lm1=Gk1Lm1/s/2;
% test for n!/2 is fulfiled
s=0;
for k=2:Lm1
   s=s+(k-1)^n*Gk1Lm1(k);
end
if s - factorial(n)/2 <.001
    disp('Gk1Lm1 is OK')
end
af=[ [5 5 5 5 5]' [0 1 2 3 4]' Gk1Lm1];
tabel_Gk1Lm1 = sprintf(% 4.0f % 4.0f %+18.14f\n',af')
```

```
save f_Gk1Lm1 Gk1Lm1
format short
return
```

Once the values of G_k^1 (5), $k \in [-4,4]$ are obtained, one can determine the values of

$$G_k^1(x), x = 0,1,2,\ldots,L-2 \text{ and } |k| \leq L-2.$$

We note there are only $(L-2)^2$ independent unknowns. Let substitute $x = 1,2,3,4$ and $k \in \mathbf{Z}, k \in [-4,4]$ into (7). One obtain an unhomogenous system with the unknowns:

$G_{-3}^1,(1), G_{-2}^1(1), G_{-1}^1(1), G_0^1(1), G_{-2}^1(2), G_{-1}^1(2), G_0^1(2), G_1^1(2),$

$G_{-1}^1,(3), G_0^1(3), G_1^1(3), G_2^1(3), G_0^1(4), G_1^1(4), G_2^1(4), G_3^1(4),$

The mentioned system has the form :

$$(I-Q)G^1 = d$$

where :

$$G^1 = [G^1(1)\ G^1(2)\ G^1(3)G^1\ (4)]';$$

$$d = [d_1 d_2 \ldots d_{16}]$$

with :

$$G^1(x) = [G_{x-4}(x)\ G_{x-3}(x)\ G_{x-2}(x)\ G_{x-1}(x)], x = 1,2,3,4;$$

$d_1 = -p_2p_0g_4 - p_3p_0g_3 - p_3p_1g_4;$
$d_2 = -p_0p_0g_4 - p_1p_0g_3 - p_1p_1g_4;$
$d_3 = 0; d_4 = 0;$
$d_5 = -p_0p_0g_4 - p_1p_0g_3 - p_2p_0g_2 - p_3p_0g_1 -$
$p_1p_1g_4 - p_2p_1g_3 - p_3p_1g_2 - p_2p_2g_4 - p_3p_2g_3 - p_3p_3g_4;$
$d_6 = -p_0p_0g_2 - p_1p_0g_1 - p_0p_1g_3 - p_1p_1g_2 - p_0p_2g_4 - p_1p_2g_3 - p_1p_3g_4;$
$d_7 = d_8 = 0;$
$d_9 = -p_0p_0g_2 - p_1p_0g_1 + p_2p_0g_0 + p_3p_0g_1 + p_4p_0g_2 + p_5p_0g_3 - p_0p_1g_3 - p_1p_1g_2 -$
$p_2p_1g_1 + p_3p_1g_0 + p_4p_1g_1 + p_5p_1g_2 - p_0p_2g_4 - p_1p_2g_3 - p_2p_2g_2 - p_3p_2g_1 -$
$p_2p_3g_3 - p_3p_3g_2 - p_2p_4g_4 - p_3p_4g_3 - p_3p_5g_4 - p_1p_3g_4;$
$d_{10} = p_0p_0g_0 + p_1p_0g_1 + p_2p_0g_2 + p_3p_0g_3 + p_4p_0g_4 - p_0p_1g_1 + p_1p_1g_0 + p_2p_1g_1 +$
$p_3p_1g_2 + p_4p_1g_3 + p_5p_1g_4 - p_0p_2g_2 - p_1p_2g_1 - p_0p_3g_3 - p_1p_3g_2 - p_0p_4g_4 -$
$p_1p_4g_3 - p_5p_1g_4;$
$d_{10} = p_0p_0g_0 + p_1p_0g_1 + p_2p_0g_2 + p_3p_0g_3 + p_4p_0g_4 - p_0p_1g_1 + p_1p_1g_0 + p_2p_1g_1 +$
$p_3p_1g_2 + p_4p_1g_3 + p_5p_1g_4 - p_0p_2g_2 - p_1p_2g_1 - p_0p_3g_3 - p_1p_3g_2 - p_0p_4g_4 -$
$p_1p_4g_3 - p_5p_1g_4;$
$d_{11} = +p_0p_0g_2 + p_0p_1g_3 + p_2p_0g_4 + p_0p_1g_1 + p_1p_1g_2 + p_2p_1g_3 + p_3p_1g_4;$
$d_{12} = +p_0p_0g_4 + p_0p_1g_3 + p_1p_1g_4;$
$d_{13} = +p_0p_0g_0 + p_0p_1g_1 + p_2p_0g_2 + p_0p_3g_3 + p_4p_0g_4 - p_0p_1g_1 + p_1p_1g_0 + p_1p_2g_1 +$
$p_3p_1g_2 + p_4p_1g_3 + p_5p_1g_4 - p_0p_2g_2 -$
$p_2p_1g_1 + p_2p_2g_0 + p_3p_2g_1 + p_4p_2g_2 + p_2p_5g_3 - p_0p_3g_3 - p_1p_3g_2 - p_2p_3g_1 +$
$p_3p_3g_0 + p_4p_1g_1 + p_5p_3g_3 - p_0p_4g_4 - p_1p_4g_3 - p_4p_2g_2 - p_3p_4g_1 - p_1p_5g_4 -$

$p_2p_5g_3 - p_3p_5g_2;$
$d_{14} = +p_0p_0g_2 + p_0p_1g_3 + p_2p_0g_4 + p_0p_1g_1 + p_1p_1g_2 + p_2p_1g_3 + p_3p_1g_4 + p_0p_2g_0 +$
$p_1p_2g_1 + p_2p_2g_2 + p_3p_2g_3 + p_4p_2g_4 - p_3p_0g_1 + p_3p_1g_0 + p_2p_3g_1 + p_3p_3g_2 +$
$p_3p_4g_3 + p_5p_3g_4 - p_0p_4g_2 - p_1p_4g_1 - p_0p_5g_3 - p_1p_5g_2;$
$d_{15} = p_0p_0g_4 + p_0p_1g_1 + p_1p_1g_4 + p_0p_3g_1 + p_1p_3g_3 + p_2p_3g_3 + p_2p_2g_4 + p_0p_2g_2 +$
$p_1p_2g_3 + p_3p_3g_4;$
$d_{16} = p_0p_2g_4 + p_0p_3g_3 + p_1p_3g_4;$

$$Q = \begin{bmatrix} Q_{1,1} & Q_{1,2} & Q_{1,3} & Q_{1,4} \\ Q_{2,1} & Q_{2,2} & Q_{2,3} & Q_{2,4} \\ Q_{3,1} & Q_{3,2} & Q_{3,3} & Q_{4,4} \\ Q_{4,1} & Q_{4,2} & Q_{4,3} & Q_{4,4} \end{bmatrix}.$$

$Q_{i,j} = [q_{i,j,k,m}]_{1 \le k,m \le L-2}; q_{i,j,m} = p_{2i-j}p_{L-1-2k+m};$
$g_0 = G_0^1(5); g_1 = G_1^1(5); g_2 = G_2^1(5);$
$g_3 = G_3^1(5); g_4 = G_4^1(5);$

The following script generates the matrices Q, d and solves the mentioned system whose solution is close of those in (8) but more accurate:

```
%p5Gk1
clc, clear all
load p_i;
load f_Gk1Lm1
load f_theta1int
Linii=[];
n=1; L=6;
Lm2=L-2;
Lm1=L-1;
Q=zeros(16);
xl=[4 2 0 -2];
for g=1:4
    for l=1:4
        %l
        li=[]; col=1;
        for i2=g*2-1:-1:g*2-4
            for i1=xl(l):1:xl(l)+3
                if i1<0 | i1>5
                    p1=0; ii1=-7;
                else
                    p1=p(i1+1); ii1=i1;
                end
                if i2>5 | i2<0
                    p2=0; ii2=-7;
                else
                    p2=p(i2+1); ii2=i2;
                end
```

```
                eps=0;
                if ii1==-7 | ii2==-7
                    ii1=0; ii2=0; eps=1;
                end
                elem= -p1*p2;
                Q((g-1)*4+l,col) =elem;
                col=col+1;
                if eps==0
                    ix=[num2str(ii1) num2str(ii2)];
                else
                    ix=[' -' ];
                end
                li=[li ix ' '];
            end
        end
        linia=(g-1)*4+l;
        if linia<10
            linia=[' ' num2str(linia)];
        else
            linia=[num2str(linia)];
        end
        lin = (['Linia ' linia '  '   li]);
        Linii=[Linii;lin];
    end
end

disp('Systems of indices of Q coefficients :')
Linii
I=eye(16);
Q=I+Q;

detQ=det(Q);
disp(['Before transformation =' num2str(detQ)])
% replace in Q row for m
m=1;
mm = m;
mmL = [m-L+2:(m-1)].^n;
Q((m-1)*4+1,:)=0;
Q((m-1)*4+1,(m-1)*4+1:(m-1)*4+4)=mmL;
detQ=det(Q);
disp(['After transformation =' num2str(det(Q))])
u=0;
v=0;
dd=zeros(Lm2^2,1);
LL=L;
```

```
for m=1:Lm2
    pmk=1;
   for k=3-LL:m-1
      s=0;
      for i=0:1:Lm1
         for j=0:1:Lm1
            if Lm1<=2*m-2*k - i | Lm1<=2*m-j
               kij=2*k+i-j;
               if  abs(kij)<=Lm2
                   if kij<0
                       temp= - Gk1Lm1(-kij+1);
                   elseif kij>0
                       temp= Gk1Lm1(kij+1);
                   else
                       temp = 0;
                   end
                   s=s+p(i+1)*p(j+1)*temp;
                   %disp([num2str(i+1)  num2str(j+1) num2str(kij)])
               end
           end
         end
      end
      mk=(m-1)*Lm2+pmk;
      pmk=pmk+1;
      dd(mk)=s;
      %mk
  end
  LL=LL-1;
end
format long

ii=[1:16]';
inl=['----------------']';
af=[ii dd]';
tab_termen_liber=sprintf('%6.0f  %14.10f\n',af)
% Replacement of right term
dd((mm-1)*4+1)=factorial(n) * theta1int(mm);
disp(' Free term after replacement : ')
af=[ii dd]';
sprintf('%6.0f  %14.10f\n',af)
% System solving
Gk1x_int=Q/dd;ii=[1:16];
%m = 1
Gk1x_int=...
[Gk1x_int; -flipud(Gk1Lm1); Gk1Lm1(2:5)];
```

```
for m=1:5
    i=1;s=0;
    for k= m-L+2:m-1, s=s+k^n * ...
        Gk1x_int((m-1)*4+i);, i=i+1;,end
 disp(['s = ' num2str(s)])
 if abs(s - factorial(n) * ...
    theta1int(m))<.1e-3
           disp(...
          ['Solution is OK!' '  m='
            num2str(m)])
 else
          disp(...
['The condition for m isn't fullfill m =' num2str(m) ',
                         theta1int(m) =' num2str(theta1int(m))])
 end
end
format short
ii=[1:25];ix=...
[ones(1,4) 2.*ones(1,4) 3.*ones(1,4)... 4*ones(1,4) 5*ones(1,9)];
d4=[4 3 2 1]; ik = ix - [d4 d4 d4 d4 [9 8 7 6 5 4 3 2 1]];
af=[ii' ix' ik' Gk1x_int];
tab_Gk1x_int = sprintf('%6.0f %6.0f %6.0f  %+16.14f\n',af')
save f_Gk1x_int Gk1x_int

The values for are :
           k     x      G^1_k(x)
-------------------------------------
     1      1     -3  -0.00957541664944
     2      1     -2  +0.24754384913050
     3      1     -1  -1.06710301814086
     4      1      0  +0.83697908740347
     5      2     -2  +0.14375092896212
     6      2     -1  -0.77134841576353
     7      2      0  +0.07530330606022
     8      2      1  +0.56499827853070
     9      3     -1  -0.74487987887380
    10      3      0  +0.00459088425691
    11      3      1  +0.73416192562073
    12      3      2  -0.12356846261008
    13      4      0  +0.00000906940912
    14      4      1  +0.74528797987869
    15      4      2  -0.14540375259546
    16      4      3  +0.01508573633831
    17      5     -4  -0.00034246575342
    18      5     -3  -0.01461187214612
```

```
19      5     -2  +0.14520547945207
20      5     -1  -0.74520547945208
21      5      0  +0.00000000000000
22      5      1  +0.74520547945208
23      5      2  -0.14520547945207
24      5      3  +0.01461187214612
25      5      4  +0.00034246575342
```

References

[1] M. Q. Chen, C. Hwang and Y. P. Shih, *The computation of wavelet-Galerkin approximation on a bounded interval*, Intl. Journal for Numerical Methods in Engineering, vol. **39**(1996), 2921-2944

[2] I. Daubechies, *Orthonormal bases of compactly supported wavelets*, Commun. Pure Appl. Math., **41**(1990), 909-996

[3] I. Daubechies, *Then Lectures on wavelets*, Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1992

[4] A. Bultheel, *Learning to swim in a sea of wavelets*, Bull. Belg. Math. Soc. **2**(1995), 1-44

[5] A. Latto and E. Tennenbaum, *Les Ondelettes a Support Compact et la Solution numerique de l'Equation de Burger*, C.R.Acad. Sci. Paris, **311**(1990), 903-909