Holistic discretisation of wave-like PDEs, II

Tony Roberts Meng Cao

January 7, 2012

1 Introduction

Try to develop good numerics of wave-like PDEs using a staggered element approach. For 'small' parameter ν, the PDEs for fields $h(x,t)$ and $u(x,t)$ are among

$$\begin{aligned}
\frac{\partial h}{\partial t} &= -\frac{\partial u}{\partial x}\,, \\
\frac{\partial u}{\partial t} &= -\frac{\partial h}{\partial x} - \nu u + \nu\frac{\partial^2 u}{\partial x^2} - \nu u\frac{\partial u}{\partial x}\,.
\end{aligned}$$

To be solved on elements centred on X_j, $X_j = jD$ say, with some coupling condition. The difference here is that we let the elements overlap so the jth element is the interval $E_j = (X_{j-1}, X_{j+1})$.

Because of even/odd symmetry I think it is more convenient to imagine two fields, each on overlapping elements, for each physical field: in element E_j introduce $h_j(x,t)$, $h'_j(x,t)$, $u_j(x,t)$ and $u'_j(x,t)$. I aim to eventuates that even-undashed fields interact with odd-undashed fields, and vice versa, but that the two sets of fields do not interact with the other. The PDEs are then

$$\begin{array}{ll}
j \text{ odd (even)} & j \text{ even (odd)} \\
\dfrac{\partial h'_j}{\partial t} = -\dfrac{\partial u_j}{\partial x}\,, & \dfrac{\partial h_j}{\partial t} = -\dfrac{\partial u'_j}{\partial x}\,, \\
\dfrac{\partial u_j}{\partial t} = -\dfrac{\partial h'_j}{\partial x} - \nu u_j\,, & \dfrac{\partial u'_j}{\partial t} = -\dfrac{\partial h_j}{\partial x} - \nu u'_j\,.
\end{array}$$

Here I propose the coupling condition on the fields, j even (odd), of

$$\begin{aligned}
&(1-\tfrac{1}{2}\gamma)\big[h_j(X_{j+1},t) - h_j(X_{j-1})\big] = \tfrac{1}{2}\gamma\big[h_{j+2}(X_{j+1},t) - h_{j-2}(X_{j-1},t)\big], \\
&u'_j(X_j,t) = \tfrac{1}{2}\big[u_{j+1}(X_j,t) + u_{j-1}(X_j,t)\big],
\end{aligned}$$

and correspondingly couple the fields, j odd (even), with

$(1-\frac{1}{2}\gamma)\big[u_j(X_{j+1},t)-u_j(X_{j-1})\big]=\frac{1}{2}\gamma\big[u_{j+2}(X_{j+1},t)-u_{j-2}(X_{j-1},t)\big]$,

$h_j'(X_j,t)=\frac{1}{2}\big[h_{j+1}(X_j,t)+h_{j-1}(X_j,t)\big]$,

Lastly, define the amplitudes to be

$$H_j=h_j(X_j)\quad\text{and}\quad U_j=u_j(X_j),$$

respectively for j even and odd (odd and even). Be careful with the dashes.

2 Eigenvalue analysis

Assume $\nu=0$, thus there is no bed drag term in the PDEs. Seek solutions in exponential form

$$u_j(x,t)=u(\xi)e^{\lambda t+ikj}\,, \tag{1}$$
$$h_j(x,t)=h(\xi)e^{\lambda t+ikj}\,, \tag{2}$$
$$u_j'(x,t)=u'(\xi)e^{\lambda t+ikj}\,, \tag{3}$$
$$h_j'(x,t)=h'(\xi)e^{\lambda t+ikj}\,, \tag{4}$$

where $\xi=(x-X_j)/D$ such that $\partial_x=\frac{1}{D}\partial_\xi$, and k is the lateral wavenumber. Substituting these exponential forms into the PDEs gives

$$\lambda^2u(\xi)=\frac{1}{D^2}\frac{\partial^2u(\xi)}{\partial\xi^2}\,,\quad\text{and}\quad\lambda^2u'(\xi)=\frac{1}{D^2}\frac{\partial^2u'(\xi)}{\partial\xi^2}\,, \tag{5}$$

which indicate solutions in the forms of

$$u(\xi)=A\cos\ell\xi+B\sin\ell\xi\,, \tag{6}$$
$$u'(\xi)=A'\cos\ell\xi+B'\sin\ell\xi\,, \tag{7}$$

where ℓ is the wavenumber of subgrid structures. Take the above $u(\xi)$ and $u'(\xi)$ into the governing PDEs again and obtain the solutions of $h(\xi)$ and $h'(\xi)$ in the forms of

$$h'(\xi)=\frac{\ell}{D\lambda}(A\sin\ell\xi-B\cos\ell\xi)\,, \tag{8}$$
$$h(\xi)=\frac{\ell}{D\lambda}(A'\sin\ell\xi-B'\cos\ell\xi)\,, \tag{9}$$

The coupling conditions proposed in the introduction indicate

$$(1-\frac{1}{2}\gamma)\big[h(1)-h(-1)\big]=\frac{1}{2}\gamma\big[h(-1)e^{2ik}-h(1)e^{-2ik}\big], \tag{10}$$
$$u'(0)=\frac{1}{2}\big[u(1)+u(-1)\big], \tag{11}$$
$$(1-\frac{1}{2}\gamma)\big[u(1)-u(-1)\big]=\frac{1}{2}\gamma\big[u(-1)e^{2ik}-u(1)e^{-2ik}\big], \tag{12}$$
$$h'(0)=\frac{1}{2}\big[h(1)+h(-1)\big], \tag{13}$$

Substitute the solutions forms of $h(\xi)$, $u(\xi)$, $h'(\xi)$ and $u'(\xi)$ into the above coupling conditions, rearrange and obtain

$$\begin{aligned}
&\big[(4-2\gamma)+\gamma(e^{2ik}+e^{-2ik})\big]\cos\ell\sin\ell\, A+\gamma\cos\ell(e^{2ik}-e^{-2ik})\, B'=0\,,\\
&-\gamma\cos\ell(e^{2ik}-e^{-2ik})\, A+\big[(4-2\gamma)+\gamma(e^{2ik}+e^{-2ik})\big]\cos\ell\sin\ell\, B'=0\,,
\end{aligned}$$

The determinant of the coefficient matrix equaling to zero gives

$$\big\{\big[(4-2\gamma)+\gamma(e^{2ik}+e^{-2ik})\big]\cos\ell\sin\ell\big\}^2=\gamma^2\cos^2\ell(e^{2ik}-e^{-2ik})^2\,, \tag{14}$$

by applying

$$e^{2ik}+e^{-2ik}=2\cos 2k \quad\text{and}\quad (e^{2ik}-e^{-2ik})^2=-4\sin^2 2k\,,$$

which becomes

$$\big[(2-\gamma)\sin\ell+\gamma\sin\ell\cos 2k\pm\gamma\sin 2k\big]\cos\ell=0\,. \tag{15}$$

Two cases arise.

1. If $\cos\ell=0$, then $\ell=\pi/2,3\pi/2,5\pi/2,\ldots$, which demonstrate the subgrid structures are good.
2. If $(2-\gamma)\sin\ell+\gamma\sin\ell\cos 2k\pm\gamma\sin 2k=0$, when $\gamma=0$, obtain $\sin\ell=0$, which indicates $\ell=0,\pi,2\pi,\ldots$. When $\gamma=1$, obtain

   $$\sin\ell+\sin\ell\cos 2k\pm\sin 2k=0\,.$$

   Triangular transforms of above equation gives

   $$(\sin\ell\cos k\pm\sin\ell)\cos k=0\,, \tag{16}$$

   which indicates a special case of $\cos k=0$ such that $k=\pi/2,3\pi/2,\ldots$, or $\sin\ell\cos k\pm\sin\ell=0$ such that

   $$\sin\ell=\pm\tan k\,. \tag{17}$$

3 Computer algebra constructs the slow manifold

Improve printing

```
on div; off allfac; on revpri;
linelength(64)$ factor dd,df;
```

Avoid slow integration with specific operator Introduce the sign function to handle the derivative discontinuities across the centre of each element. Define the integral operator to handle polynomials with sign functions, both indefinite ($\int_0^\xi d\xi$) and definite to $\xi = \mathtt{q} = \pm 1$ ($\int_0^q d\xi$).

```
operator intx; linear intx;
let { intx(xi^~~p,xi)=>xi^(p+1)/(p+1)
    , intx(1,xi)=>xi
    , intx(xi^~~p,xi,~q)=>q^(p+1)/(p+1)
    , intx(1,xi,~q)=>q
    };
```

Introduce subgrid variable Introduced above is the subgrid variable $\xi = (x - X_j)/D$, $|\xi| < 1$, in which the fields are described.

```
depend xi,x;  let df(xi,x)=>1/dd;
```

Define evolving amplitudes Amplitudes are as $U_j(t) = u_j'(X_j, t)$ and $H_j(t) = h_j'(X_j, t)$, The difference here is that we take, say, even j to be the u-elements and odd j to be the h-elements. Actually it should not matter which way around, or even if you regard the modelling as being of two disjoint systems (one one way and one the other). The amplitudes depend upon time according to some approximation stored in `gh` and `gu`.

```
operator hh; operator uu;
depend hh,t; depend uu,t;
let { df(hh(~k),t)=>sub(j=k,gh)
    , df(uu(~k),t)=>sub(j=k,gu)
    };
```

But solvability condition is coupled Now the evolution equations are coupled together. By some symmetry we decouple the equations using this operator `ginv`. However, I expect that some problems will not decouple (look for non-cancelling pollution by `ginv` operators). In which case we have to accept that the DEs for the amplitudes are *implicit* DEs using the following operator. Let's define $\mathcal{G} = E + E^{-1}$ so that $\mathcal{G}F_j = F_{j+1} + F_{j-1}$. Take $\mathcal{G}^{-1}$ of this equation to deduce $\mathcal{G}^{-1}F_{j\pm1} = F_j - \mathcal{G}^{-1}F_{j\mp1}$, and change subscripts, $j \mapsto k \mp 1$, to deduce $\mathcal{G}^{-1}F_k = F_{k\mp1} - \mathcal{G}^{-1}F_{k\mp2}$. That is, we change an inverse of $\mathcal{G}$ to one with subscript closer to $k = j$, or otherwise if we desire. Have here coded some quadratic transformations so we can resolve quadratic terms in the model, but I guess we also might want cubic.

The following causes a warning that `~a` and `~b` are declared operator, which is fine, but I cannot predefine them as operators so cannot avoid the warning.

```
operator ginv; linear ginv;
let { df(ginv(~a,t),t)=>ginv(df(a,t),t)
    , ginv(~a(j+~k),t)=>a(j+k-1)-ginv(a(j+k-2),t) when k>1
    , ginv(~a(j+~k),t)=>a(j+k+1)-ginv(a(j+k+2),t) when k<0
    , ginv(~a(j+~k)^2,t)=>a(j+k-1)^2-ginv(a(j+k-2)^2,t) wher
    , ginv(~a(j+~k)^2,t)=>a(j+k+1)^2-ginv(a(j+k+2)^2,t) wher
    , ginv(~a(j+~~k)*~b(j+~~l),t)=> a(j+k-1)*b(j+l-1)
      -ginv(a(j+k-2)*b(j+l-2),t) when k+l>2
    , ginv(~a(j+~~k)*~b(j+~~l),t)=> a(j+k+1)*b(j+l+1)
      -ginv(a(j+k+2)*b(j+l+2),t) when k+l<-1
    };
```

Start with linear approximation The linear approximation is the usual piecewise constant fields in each element. Except that the dashed fields are (surprisingly sensible) averages of the surrounding elements.

```
hj:=hh(j); hdj:=(hh(j+1)+hh(j-1))/2;
uj:=uu(j); udj:=(uu(j+1)+uu(j-1))/2;
gh:=gu:=0;
```

Truncate the asymptotic series in coupling γ and any other parameter, such as ν. The basic slow manifold model evolution only appears at odd powers of γ, so choosing errors to be even power of γ is good.

```
let gam^6=>0; factor gam;
gamma:=gam;
let nu^2=>0; factor nu;
```

Iterate to a slow manifold Iterate to seek a solution, terminating only when residuals are zero to specified order.

```
for it:=1:9 do begin
write "ITERATION = ",it;
```

Choose this order of updating fields from residuals due to the pattern of communication.

First do the equations for the evolution of the dashed fields. j even

```
resud:=df(udj,t)+df(hj,x)+nu*udj-nu*df(udj,x,2);
write lengthresud:=length(resud);
reshb:=(1-gamma/2)*(sub(xi=+1,hj)-sub(xi=-1,hj))
    -gamma/2*(sub({j=j+2,xi=-1},hj)-sub({j=j-2,xi=+1},hj));
write lengthreshb:=length(reshb);
write
gu:=gu+(gud:=ginv(reshb/dd
    -intx(resud,xi,1)+intx(resud,xi,-1),t));
hj:=hj-dd*intx(resud+sub(j=j-1,gud)/2+sub(j=j+1,gud)/2,xi);
```

j odd

```
reshd:=df(hdj,t)+df(uj,x);
write lengthreshd:=length(reshd);
resub:=(1-gamma/2)*(sub(xi=+1,uj)-sub(xi=-1,uj))
    -gamma/2*(sub({j=j+2,xi=-1},uj)-sub({j=j-2,xi=+1},uj));
write lengthresub:=length(resub);
write
gh:=gh+(ghd:=ginv(resub/dd
    -intx(reshd,xi,1)+intx(reshd,xi,-1),t));
uj:=uj-dd*intx(reshd+sub(j=j-1,ghd)/2+sub(j=j+1,ghd)/2,xi);
```

Second do the equations for the evolution of the undashed fields, to get spatial structure of dashed fields. j even

```
resh:=df(hj,t)+df(udj,x);
write lengthresh:=length(resh);
resua:=-sub(xi=0,udj)
    +sub({j=j+1,xi=-1},uj)/2+sub({j=j-1,xi=+1},uj)/2;
write lengthresua:=length(resua);
udj:=udj+resua-dd*int(resh,xi);
```

j odd

```
resu:=df(uj,t)+df(hdj,x)+nu*uj-nu*df(uj,x,2);
write lengthresu:=length(resu);
resha:=-sub(xi=0,hdj)
    +sub({j=j+1,xi=-1},hj)/2+sub({j=j-1,xi=+1},hj)/2;
write lengthresha:=length(resha);
hdj:=hdj+resha-dd*intx(resu,xi);
```

Terminate the loop Exit the loop if all residuals are zero.

```
  if {resh,reshd,resha,reshb,resu,resud,resua,resub}
  ={0,0,0,0,0,0,0,0} then write it:=it+100000;
  showtime;
end;
```

Equivalent PDEs Finish by finding the equivalent PDE for the discretisation. Since $\mathcal{G} = E + E^{-1} = e^{D\partial} + e^{-D\partial} = 2\cosh(D\partial)$ so $\mathcal{G}^{-1} = \frac{1}{2}\operatorname{sech}(D\partial)$. Find the discretisation is consistent to an order in grid spacing D that increases with order of coupling γ.

```
let dd^8=>0;
depend uu,x; depend hh,x;
rules:={uu(j)=>uu, uu(j+~p)=>uu+(for n:=1:8 sum
                 df(uu,x,n)*(dd*p)^n/factorial(n))
        ,hh(j)=>hh, hh(j+~p)=>hh+(for n:=1:8 sum
                 df(hh,x,n)*(dd*p)^n/factorial(n))
        ,ginv(~a,t)=>1/2*(a-1/2*dd^2*df(a,x,2)
        +5/24*dd^4*df(a,x,4) -61/720*dd^6*df(a,x,6)
        +277/8064*dd^8*df(a,x,8) )
        }$
ghde:=(gh where rules);
gude:=(gu where rules);
```

Draw graph of subgrid field The first plot call is a dummy that appears needed on my system for some unknown reason.

```
plot(sin(xi),terminal=aqua);
u0:=sub(j=0,uj)$ u1:=sub(j=1,udj)$
u0:=(u0 where {nu=>0,gam=>1,uu(0)=>1,uu(~k)=>0 when k neq 0
u1:=(u1 where {nu=>0,gam=>1,uu(0)=>1,uu(~k)=>0 when k neq 0
plot({u0,u1},xi=(-4 .. 4),terminal=aqua);
```

Finish

```
end;
```

4 Sample output

```
1: in_tex "waveOverRed.tex"$

*** ~a declared operator

*** ~b declared operator

hj := hh(j)

        1                 1
hdj := ---*hh(1 + j) + ---*hh( - 1 + j)
        2                 2

uj := uu(j)
```

```

        1               1
udj := ---*uu(1 + j) + ---*uu( - 1 + j)
        2               2

gh := gu := 0

gamma := gam

ITERATION = 1

lengthresud := 3

lengthreshb := 3

        -1          1               1
gu := dd  *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) - nu*uu
                    2               2

lengthreshd := 1

lengthresub := 3

        -1          1               1
gh := dd  *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j))
                    2               2

lengthresh := 7

lengthresua := 5

lengthresu := 7

lengthresha := 5

Time: 20 ms

ITERATION = 2

lengthresud := 12

lengthreshb := 5

        -1          1               1                    -1
gu := dd  *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) + dd  *
                    2               2
```

```

            1                1                1
      *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 +
            16               48               16

            1
         - ----*hh( - 3 + j)) - nu*uu(j)
            48

lengthreshd := 12

lengthresub := 5

        -1          1               1                    -1
gh := dd  *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j)) + dd  *
                    2               2

            1                1                1
      *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 +
            16               48               16

            1
         - ----*uu( - 3 + j))
            48

lengthresh := 15

lengthresua := 5

lengthresu := 15

lengthresha := 5

Time: 10 ms

ITERATION = 3

lengthresud := 7

lengthreshb := 7

        -1          1               1                    -1
gu := dd  *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) + dd  *
                    2               2

            1                1                1
```

```
      *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 +
            16               48               16

            1
         - ----*hh( - 3 + j)) - nu*uu(j)
            48

lengthreshd := 7

lengthresub := 7

        -1          1               1                    -1
gh := dd  *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j)) + dd  *
                    2               2

            1                1                1
      *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 +
            16               48               16

            1
         - ----*uu( - 3 + j))
            48

lengthresh := 1

lengthresua := 9

lengthresu := 1

lengthresha := 9

Time: 10 ms

ITERATION = 4

lengthresud := 1

lengthreshb := 1

        -1          1               1                    -1
gu := dd  *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) + dd  *
                    2               2

            1                1                1
      *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 +
            16               48               16
```

```
            1
         - ----*hh( - 3 + j)) - nu*uu(j)
            48

lengthreshd := 1

lengthresub := 1

        -1          1               1                    -1
gh := dd  *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j)) + dd  *
                    2               2

            1                1                1
      *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 +
            16               48               16

            1
         - ----*uu( - 3 + j))
            48

lengthresh := 1

lengthresua := 1

lengthresu := 1

lengthresha := 1

it := 100004

Time: 10 ms

                           1               2
ghde :=  - df(uu,x)*gam - ---*df(uu,x,3)*dd *gam
                           6

            1               2    3     1                4
         + ---*df(uu,x,3)*dd *gam  - -----*df(uu,x,5)*dd *ga
            6                         120

            1                4    3     1                 6
         + ----*df(uu,x,5)*dd *gam  - ------*df(uu,x,7)*dd *
            12                         5040

            13                6    3
```

```
         + -----*df(uu,x,7)*dd *gam
            720

                                   1               2
gude :=  - nu*uu - df(hh,x)*gam - ---*df(hh,x,3)*dd *gam
                                   6

            1               2    3     1               4
         + ---*df(hh,x,3)*dd *gam  - -----*df(hh,x,5)*dd *ga
            6                         120

            1                4    3     1                 6
         + ----*df(hh,x,5)*dd *gam  - ------*df(hh,x,7)*dd *
            12                         5040

            13               6    3
         + -----*df(hh,x,7)*dd *gam
            720
```