Holistic discretisation of wave-like PDEs, II

Tony Roberts   Meng Cao

January 19, 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

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

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

$$
\begin{array}{cc}
j \text{ odd (even)} & j \text{ even (odd)} \\[4pt]
\dfrac{\partial h'_j}{\partial t} = -\dfrac{\partial u_j}{\partial x}, & \dfrac{\partial h_j}{\partial t} = -\dfrac{\partial u'_j}{\partial x}, \\[10pt]
\dfrac{\partial u_j}{\partial t} = -\dfrac{\partial h'_j}{\partial x} - \nu u_j, & \dfrac{\partial u'_j}{\partial t} = -\dfrac{\partial h_j}{\partial x} - \nu u'_j.
\end{array}
$$

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

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

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

$$(1 - \tfrac{1}{2}\gamma)\big[u_j(X_{j+1}, t) - u_j(X_{j-1}, t)\big] = \tfrac{1}{2}\gamma\big[u_{j+2}(X_{j+1}, t) - u_{j-2}(X_{j-1}, t)\big],$$
$$h'_j(X_j, t) = \tfrac{1}{2}\big[h_{j+1}(X_j, t) + h_{j-1}(X_j, t)\big],$$

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 dissipation (bed drag) 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 $\xi = (x - X_j)/D$ so that $\partial_x = \frac{1}{D}\partial_\xi$, and k is the lateral wavenumber. In essence, $u(\xi), h(\xi), u'(\xi), h'(\xi)$ are Fourier transforms over the element index j of the corresponding fields.

Use my command \DD to do second order partial derivatives, and \D for first order partial derivative.

Substituting these exponential forms into the PDEs gives

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

Use one line if reasonable.

Being constant coefficient we try solutions for the subgrid structure in terms of trigonometric functions:

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

where ℓ is the wavenumber of the subgrid structures. Substitute the above $u(\xi)$ and $u'(\xi)$ into the PDEs (5) and obtain that the solutions $h(\xi)$ and $h'(\xi)$ must take the forms

Always be more explicit: cross-reference precisely, not vaguely.

$$h'(\xi) = \frac{\ell}{D\lambda}(A\sin\ell\xi - B\cos\ell\xi) \quad\text{and}\quad h(\xi) = \frac{\ell}{D\lambda}(A'\sin\ell\xi - B'\cos\ell\xi). \tag{7}$$

The coupling conditions (**??**) proposed in the introduction in-

dicate

$$(1 - \frac{1}{2}\gamma)\big[h(1) - h(-1)\big] = \frac{1}{2}\gamma\big[h(-1)e^{2ik} - h(1)e^{-2ik}\big], \quad (8)$$

$$u'(0) = \frac{1}{2}\big[u(1) + u(-1)\big], \quad (9)$$

$$(1 - \frac{1}{2}\gamma)\big[u(1) - u(-1)\big] = \frac{1}{2}\gamma\big[u(-1)e^{2ik} - u(1)e^{-2ik}\big], \quad (10)$$

$$h'(0) = \frac{1}{2}\big[h(1) + h(-1)\big], \quad (11)$$

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

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

Nontrivial solutions of these equations exist only when the coefficient matrix is singular. Setting the determinant of the coefficient matrix equaling to zero gives the characteristic equation

$$\big\{\big[(4 - 2\gamma) + \gamma(e^{2ik} + e^{-2ik})\big]\cos\ell\sin\ell\big\}^2 = \gamma^2\cos^2\ell(e^{2ik} - e^{-2ik})^2\,. \quad (12)$$

Invoking

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

the characteristic equation becomes

$$\big[(2 - \gamma)\sin\ell + \gamma\sin\ell\cos 2k \pm \gamma\sin 2k\big]\cos\ell = 0\,. \quad (13)$$

Two possibilities arise.

1. If $\cos\ell = 0$, then $\ell = \pi/2, 3\pi/2, 5\pi/2, \ldots$ for all coupling γ.

2. If $(2 - \gamma)\sin\ell + \gamma\sin\ell\cos 2k \pm \gamma\sin 2k = 0$, in the decoupled case, $\gamma = 0$, obtain $\sin\ell = 0$, which indicates $\ell = 0, \pi, 2\pi, \ldots$. In the case of full coupling, $\gamma = 1$, obtain

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

Triangular transforms of above equation gives

$$(\sin\ell\cos k \pm \sin k)\cos k = 0\,, \quad (14)$$

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

$$\sin\ell = \pm\tan k\,. \quad (15)$$

Margin annotations:

Give a couple of words to describe what is coming, namely, here two equations

You have missed out some steps here as B and A' have disappeared. Describe.

Use more words.

Avoid "$\backslash[\cdots\backslash]$", use either `displaymath` or `equation*` environments.

Need some interpretation.

Need some interpretation.

Now, as well as discussing the above further, the other thing to do here is to check that the expansion in small coupling γ of the characteristic equation (13)

3 Computer algebra constructs the slow manifold

Improve printing

```
1 on div; off allfac; on revpri;
2 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 $\xi = \mathtt{q} = \pm 1$ ($\int_0^q d\xi$).

```
3 operator intx; linear intx;
4 let { intx(xi^~~p,xi)=>xi^(p+1)/(p+1)
5     , intx(1,xi)=>xi
6     , intx(xi^~~p,xi,~q)=>q^(p+1)/(p+1)
7     , intx(1,xi,~q)=>q
8     };
```

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

```
9 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`.

```
10 operator hh; operator uu;
11 depend hh,t; depend uu,t;
12 let { df(hh(~k),t)=>sub(j=k,gh)
13     , df(uu(~k),t)=>sub(j=k,gu)
14     };
```

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.

```
15 operator ginv; linear ginv;
16 let { df(ginv(~a,t),t)=>ginv(df(a,t),t)
17     , ginv(~a(j+~k),t)=>a(j+k-1)-ginv(a(j+k-2),t) when k>1
18     , ginv(~a(j+~k),t)=>a(j+k+1)-ginv(a(j+k+2),t) when k<0
19     , ginv(~a(j+~k)^2,t)=>a(j+k-1)^2-ginv(a(j+k-2)^2,t) when
20     , ginv(~a(j+~k)^2,t)=>a(j+k+1)^2-ginv(a(j+k+2)^2,t) when
21     , ginv(~a(j+~~k)*~b(j+~~l),t)=> a(j+k-1)*b(j+l-1)
22       -ginv(a(j+k-2)*b(j+l-2),t) when k+l>2
23     , ginv(~a(j+~~k)*~b(j+~~l),t)=> a(j+k+1)*b(j+l+1)
24       -ginv(a(j+k+2)*b(j+l+2),t) when k+l<-1
25     };
```

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.

```
26 hj:=hh(j); hdj:=(hh(j+1)+hh(j-1))/2;
27 uj:=uu(j); udj:=(uu(j+1)+uu(j-1))/2;
28 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.

```
29 let gam^6=>0; factor gam;
30 gamma:=gam;
31 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.

```
32 for it:=1:9 do begin
33 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

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

j odd

```
43 reshd:=df(hdj,t)+df(uj,x);
44 write lengthreshd:=length(reshd);
45 resub:=(1-gamma/2)*(sub(xi=+1,uj)-sub(xi=-1,uj))
46     -gamma/2*(sub({j=j+2,xi=-1},uj)-sub({j=j-2,xi=+1},uj));
47 write lengthresub:=length(resub);
48 write
49 gh:=gh+(ghd:=ginv(resub/dd
50     -intx(reshd,xi,1)+intx(reshd,xi,-1),t));
51 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

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

j odd

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

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

```
64    if {resh,reshd,resha,reshb,resu,resud,resua,resub}
65    ={0,0,0,0,0,0,0,0} then write it:=it+100000;
66    showtime;
67 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.

```
68 let dd^8=>0;
69 depend uu,x; depend hh,x;
70 rules:={uu(j)=>uu, uu(j+~p)=>uu+(for n:=1:8 sum
71               df(uu,x,n)*(dd*p)^n/factorial(n))
72        ,hh(j)=>hh, hh(j+~p)=>hh+(for n:=1:8 sum
73               df(hh,x,n)*(dd*p)^n/factorial(n))
74        ,ginv(~a,t)=>1/2*(a-1/2*dd^2*df(a,x,2)
75        +5/24*dd^4*df(a,x,4) -61/720*dd^6*df(a,x,6)
76        +277/8064*dd^8*df(a,x,8) )
77        }$
78 ghde:=(gh where rules);
79 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.

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

Finish

```
85 end;
```

4   Sample output

```
86 1: in_tex "waveOverRed.tex"$
87
88 *** ~a declared operator
89
90 *** ~b declared operator
91
```

```
 92 hj := hh(j)
 93
 94         1                1
 95 hdj := ---*hh(1 + j) + ---*hh( - 1 + j)
 96         2                2
 97
 98 uj := uu(j)
 99
100         1                1
101 udj := ---*uu(1 + j) + ---*uu( - 1 + j)
102         2                2
103
104 gh := gu := 0
105
106 gamma := gam
107
108 ITERATION = 1
109
110 lengthresud := 3
111
112 lengthreshb := 3
113
114          -1              1                1
115 gu := dd   *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) - nu*uu
116                      2                2
117
118 lengthreshd := 1
119
120 lengthresub := 3
121
122          -1              1                1
123 gh := dd   *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j))
124                      2                2
125
126 lengthresh := 7
127
128 lengthresua := 5
129
130 lengthresu := 7
131
132 lengthresha := 5
133
134 Time: 20 ms
135
136 ITERATION = 2
137
```

```
138 lengthresud := 12
139
140 lengthreshb := 5
141
142          -1              1                   1                         -1
143 gu := dd  *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) + dd   *
144                      2               2
145
146            1              1              1
147       *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 +
148            16             48             16
149
150            1
151          - ----*hh( - 3 + j)) - nu*uu(j)
152            48
153
154 lengthreshd := 12
155
156 lengthresub := 5
157
158          -1              1                   1                         -1
159 gh := dd  *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j)) + dd   *
160                      2               2
161
162            1              1              1
163       *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 +
164            16             48             16
165
166            1
167          - ----*uu( - 3 + j))
168            48
169
170 lengthresh := 15
171
172 lengthresua := 5
173
174 lengthresu := 15
175
176 lengthresha := 5
177
178 Time: 10 ms
179
180 ITERATION = 3
181
182 lengthresud := 7
183
```

```
184 lengthreshb := 7
185
186           -1              1                1                      -1
187 gu := dd  *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) + dd   *
188                     2                2
189
190              1                1               1
191      *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 +
192          16               48               16
193
194              1
195          - ----*hh( - 3 + j)) - nu*uu(j)
196            48
197
198 lengthreshd := 7
199
200 lengthresub := 7
201
202           -1              1                1                      -1
203 gh := dd  *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j)) + dd   *
204                     2                2
205
206              1                1               1
207      *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 +
208          16               48               16
209
210              1
211          - ----*uu( - 3 + j))
212            48
213
214 lengthresh := 1
215
216 lengthresua := 9
217
218 lengthresu := 1
219
220 lengthresha := 9
221
222 Time: 10 ms
223
224 ITERATION = 4
225
226 lengthresud := 1
227
228 lengthreshb := 1
229
```

10

```
230          -1           1              1                    -1
231 gu := dd  *gam*( - ---*hh(1 + j) + ---*hh( - 1 + j)) + dd  *
232                     2              2

233

234           1              1              1
235     *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 +
236          16             48             16

237

238           1
239        - ----*hh( - 3 + j)) - nu*uu(j)
240          48

241

242 lengthreshd := 1

243

244 lengthresub := 1

245

246          -1           1              1                    -1
247 gh := dd  *gam*( - ---*uu(1 + j) + ---*uu( - 1 + j)) + dd  *
248                     2              2

249

250           1              1              1
251     *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 +
252          16             48             16

253

254           1
255        - ----*uu( - 3 + j))
256          48

257

258 lengthresh := 1

259

260 lengthresua := 1

261

262 lengthresu := 1

263

264 lengthresha := 1

265

266 it := 100004

267

268 Time: 10 ms

269

270                             1                2
271 ghde :=  - df(uu,x)*gam - ---*df(uu,x,3)*dd *gam
272                           6

273

274           1                2   3    1                    4
275        + ---*df(uu,x,3)*dd *gam  - -----*df(uu,x,5)*dd *ga
```

11

```
276           6                           120

277

278           1                   4   3   1                      6
279     + ----*df(uu,x,5)*dd *gam  - ------*df(uu,x,7)*dd *
280          12                       5040

281

282          13                  6   3
283     + -----*df(uu,x,7)*dd *gam
284         720

285

286                                   1                   2
287 gude :=  - nu*uu - df(hh,x)*gam - ---*df(hh,x,3)*dd *gam
288                                   6

289

290           1                   2   3   1                      4
291     + ---*df(hh,x,3)*dd *gam  - -----*df(hh,x,5)*dd *ga
292          6                       120

293

294           1                   4   3   1                      6
295     + ----*df(hh,x,5)*dd *gam  - ------*df(hh,x,7)*dd *
296          12                       5040

297

298          13                  6   3
299     + -----*df(hh,x,7)*dd *gam
300         720
```