Holistic discretisation of wave-like PDEs, II

Tony Roberts Meng Cao

February 24, 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

$$\begin{aligned} &j \text{ odd (even)} && j \text{ even (odd)} && (1)\\ &\frac{\partial h'_j}{\partial t} = -\frac{\partial u_j}{\partial x}\,, && \frac{\partial h_j}{\partial t} = -\frac{\partial u'_j}{\partial x}\,, && (2)\\ &\frac{\partial u_j}{\partial t} = -\frac{\partial h'_j}{\partial x} - \nu u_j\,, && \frac{\partial u'_j}{\partial t} = -\frac{\partial h_j}{\partial x} - \nu u'_j\,. && (3) \end{aligned}$$

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

$$(1-\tfrac{1}{2}\gamma)\big[h_j(X_{j+1},t) - h_j(X_{j-1},t)\big] = \tfrac{1}{2}\gamma\big[h_{j+2}(X_{j+1},t) - h_{j-2}(X_{j-1},t)\big], \tag{4}$$

$$u'_j(X_j,t) = \tfrac{1}{2}\big[u_{j+1}(X_j,t) + u_{j-1}(X_j,t)\big], \tag{5}$$

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

$$(1-\tfrac{1}{2}\gamma)\big[u_j(X_{j+1},t) - u_j(X_{j-1})\big] = \tfrac{1}{2}\gamma\big[u_{j+2}(X_{j+1},t) - u_{j-2}(X_{j-1},t)\big], \tag{6}$$

$$h'_j(X_j,t) = \tfrac{1}{2}\big[h_{j+1}(X_j,t) + h_{j-1}(X_j,t)\big], \tag{7}$$

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—or maybe ditch dashes as in the following section.

2 ANZIAM talk

This section possibly provides some nice pictures to use.

2.1 Challenge: couple patches of simulations

See gapToothWave.mpg

- given some complicated microscale simulator,
- that is far too expensive to use over a large domain,
- simulate only on *small* patches in space.

Challenge how do we couple such patchy simulators across 'empty' space?

2.2 Explore the wave equation

The microscale simulator could be complicated such as a model of interacting ice pieces on the surface of water (growlers or bergy bits), or a turbulent model of tsunamis and floods.

The macroscale could be tides or long water waves, wavelength $> 100\,\mathrm{m}$ say.

Challenge couple, across gaps, such wave-like simulators?

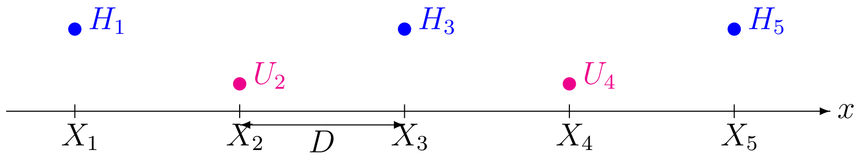
Here start with just the canonical wave equation for complementary fields $h(x,t)$ and $u(x,t)$

$$\frac{\partial h}{\partial t} = -\frac{\partial u}{\partial x}\,, \qquad \frac{\partial u}{\partial t} = -\frac{\partial h}{\partial x}\,.$$

But aim for when we do not know the precise form of the RHS*s.*

2.3 Staggered macroscale and microscale

Classic discrete approximation for a wave equation is on a staggered grid and has lots of good properties:

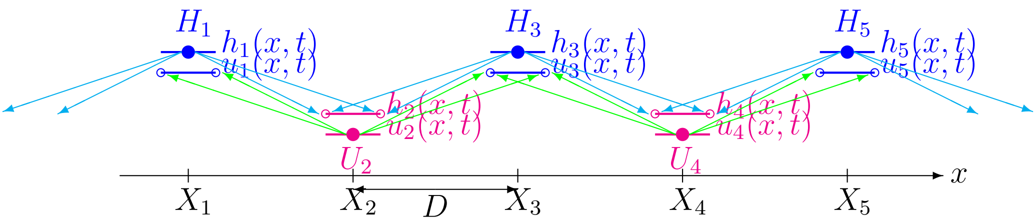


$$\begin{aligned} \text{odd } j \quad \frac{\partial H_j}{\partial t} &= -\frac{U_{j+1}-U_{j-1}}{2D}\,, \\ \text{even } j \quad \frac{\partial U_j}{\partial t} &= -\frac{H_{j+1}-H_{j-1}}{2D}\,. \end{aligned}$$

But suppose we do not know the RHSs algebraically, only simulator.

Simulations used a staggered macroscale grid of patches.

Each patch around a macroscale grid point X_j executes a microscale simulator of the microscale $h_j(x,t)$ and $u_j(x,t)$:

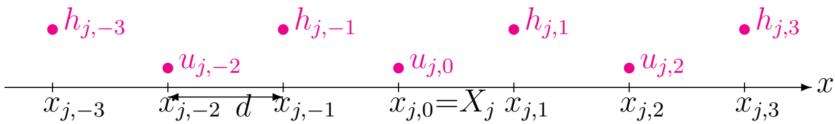


odd j interpolating $U_{j\pm1}$ gives BCs $u_j(x,t)$ gives $H_j(t)$;

even j interpolating $H_{j\pm1}$ gives BCs $h_j(x,t)$ gives $U_j(t)$.

Microscale simulation movie used a staggered microscale grid.

For example, zoom onto an even j patch around a macroscale grid point:

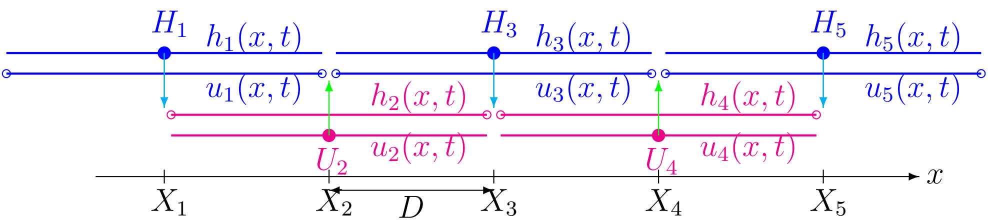


- odd i, $\partial h_{j,i}/\partial t = -\frac{u_{j,i+1}-u_{j,i-1}}{2d}$,
- even i, $\partial u_{j,i}/\partial t = -\frac{h_{j,i+1}-h_{j,i-1}}{2d}$,
- boundary $h_{j,\pm 3}$ from interpolating $H_{j\pm 1}$ (potential $H_{j\pm 3}, \ldots$)
- define macroscale $U_j(t) = u_{j,0}(t)$.

This scheme generated the movie.

2.4 Slow manifold from overlapping elements!

Following beautiful modelling of parabolic reaction-advection-diffusion equations, interpolating macroscale values easiest when elements overlap!



No interpolation required! Coupling boundary conditions are

- even j, $h_j(X_{j\pm 1}, t) = H_{j\pm 1} := h_{j\pm 1}(X_{j\pm 1}, t)$
- odd j, $u_j(X_{j\pm 1}, t) = U_{j\pm 1} := u_{j\pm 1}(X_{j\pm 1}, t)$

Embed the wave dynamics in a family introduce a coupling parameter γ, $0 \le \gamma \le 1$, to range from uncoupled elements, $\gamma = 0$, to fully coupled $\gamma = 1$.

- even j, $h_j(X_{j\pm1},t)=\gamma h_{j\pm1}(X_{j\pm1},t)+(1-\gamma)h_j(X_j,t)$
- odd j, $u_j(X_{j\pm1},t)=\gamma u_{j\pm1}(X_{j\pm1},t)+(1-\gamma)u_j(X_j,t)$

Find a slow subspace when decoupled, $\gamma=0$, that is robust to perturbations and predicts slow manifold evolution when fully coupled, $\gamma=1$.

The slow manifold is the macroscale modes.

Fast oscillations about the slow manifold are microscale modes.

Find the slow manifold Substitute fields $u_j,h_j\propto\exp[i(kj+\ell t/D)]$ for macroscale variations of wavenumber k and frequency ℓ/D.

Microscale: seek solutions trigonometric in $\ell(x-X_j)/D$, microscale wavenumber ℓ, … find the characteristic equation

$$\big[(2-\gamma)\sin\ell+\gamma\sin\ell\cos2k\pm\gamma\sin2k\big]\cos\ell=0\,.$$

Real frequency $\ell(k,\gamma)$ implies no instability (as in movie).

Small coupling γ asymptotics gives equivalent macroscale PDE:

$$\begin{aligned}\frac{\partial H}{\partial t}&=-\gamma\frac{\partial U}{\partial x}-\tfrac16(\gamma-\gamma^2)D^2\frac{\partial^3U}{\partial x^3}-\tfrac1{120}(\gamma-10\gamma^2)D^4\frac{\partial^5U}{\partial x^5}+\mathcal O\big(\gamma^3\big)\\ \frac{\partial U}{\partial t}&=-\gamma\frac{\partial H}{\partial x}-\tfrac16(\gamma-\gamma^2)D^2\frac{\partial^3H}{\partial x^3}-\tfrac1{120}(\gamma-10\gamma^2)D^4\frac{\partial^5H}{\partial x^5}+\mathcal O\big(\gamma^3\big)\end{aligned}$$

2.5 Conclusion

- A macroscale staggered grid of microscale patches appears to be an efficient way to simulate wave-like systems.
- Linear analysis of overlapping patches supports the modelling as a slow, sub-centre, manifold.

- Future linear analysis will explore small patches, not overlapping, as seen in numerical simulations.
- Then support for modelling nonlinear dynamics automatically follows for future applications.

3 Eigenvalue analysis

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

$$u_j(x,t) = u(\xi)e^{\lambda t + ikj}\,, \tag{8}$$
$$h_j(x,t) = h(\xi)e^{\lambda t + ikj}\,, \tag{9}$$
$$u'_j(x,t) = u'(\xi)e^{\lambda t + ikj}\,, \tag{10}$$
$$h'_j(x,t) = h'(\xi)e^{\lambda t + ikj}\,, \tag{11}$$

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. Substituting these exponential forms into the PDEs gives

$$\lambda^2 u(\xi) = \frac{1}{D^2}\frac{\partial^2 u(\xi)}{\partial \xi^2} \quad\text{and}\quad \lambda^2 u'(\xi) = \frac{1}{D^2}\frac{\partial^2 u'(\xi)}{\partial \xi^2}\,. \tag{12}$$

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

$$u(\xi) = A\cos\ell\xi + B\sin\ell\xi \quad\text{and}\quad u'(\xi) = A'\cos\ell\xi + B'\sin\ell\xi\,, \tag{13}$$

where ℓ is the wavenumber of the subgrid structures. Substituting the above $u(\xi)$ and $u'(\xi)$ into the PDEs (12) indicates

$$\lambda^2 = -\frac{\ell^2}{D^2}\,, \tag{14}$$

and the solutions $h(\xi)$ and $h'(\xi)$ must take the forms

$$h'(\xi) = \frac{\ell}{D\lambda}(A\sin\ell\xi - B\cos\ell\xi) \quad \text{and} \quad h(\xi) = \frac{\ell}{D\lambda}(A'\sin\ell\xi - B'\cos\ell\xi)\,. \tag{15}$$

The coupling conditions (4)–(7) proposed in the introduction indicate

$$(1-\frac{1}{2}\gamma)\big[h(1)-h(-1)\big] = \frac{1}{2}\gamma\big[h(-1)e^{2ik} - h(1)e^{-2ik}\big], \tag{16}$$

$$u'(0) = \frac{1}{2}\big[u(0)e^{ik} + u(0)e^{-ik}\big], \tag{17}$$

$$(1-\frac{1}{2}\gamma)\big[u(1)-u(-1)\big] = \frac{1}{2}\gamma\big[u(-1)e^{2ik} - u(1)e^{-2ik}\big], \tag{18}$$

$$h'(0) = \frac{1}{2}\big[h(0)e^{ik} + h(0)e^{-ik}\big], \tag{19}$$

Substitute the solutions forms (13) and (15) of $h(\xi)$, $u(\xi)$, $h'(\xi)$ and $u'(\xi)$ into the above coupling conditions (16)–(19), obtain

$$\begin{aligned}&(1-\frac{1}{2}\gamma)\left[\frac{\ell}{D\lambda}(A'\sin\ell - B'\cos\ell + A'\sin\ell + B'\cos\ell)\right]\\ &\quad = \frac{1}{2}\gamma\frac{\ell}{D\lambda}\left[(-A'\sin\ell - B'\cos\ell)e^{2ik} - (A'\sin\ell - B'\cos\ell)e^{-2ik}\right]\,,\end{aligned} \tag{20}$$

$$A' = \frac{1}{2}A(e^{ik}+e^{-ik})\,, \tag{21}$$

$$\begin{aligned}&(1-\frac{1}{2}\gamma)(A\cos\ell + B\sin\ell - A\cos\ell + B\sin\ell)\\ &\quad = \frac{1}{2}\gamma\left[(A\cos\ell - B\sin\ell)e^{2ik} - (A\cos\ell + B\sin\ell)e^{-2ik}\right]\,,\end{aligned} \tag{22}$$

$$B = \frac{1}{2}B'(e^{ik}+e^{-ik})\,, \tag{23}$$

by replacing A' and B and applying $e^{ik}+e^{-ik} = 2\cos k$, which gives the following two equations of

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

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

$$\left[(4-2\gamma)+\gamma(e^{2ik}+e^{-2ik})\right]\sin^2\ell\cos^2k+\gamma^2\cos^2\ell(e^{2ik}-e^{-2ik})^2=0\,. \quad (24)$$

Invoking

$$\begin{aligned} e^{2ik}+e^{-2ik}=2\cos 2k \quad &\text{and} \quad (e^{2ik}-e^{-2ik})^2=-4\sin^2 2k\,, \\ \cos 2k=1-2\sin^2 k \quad &\text{and} \quad \sin^2 2k=4\sin^k\cos^2 k\,, \end{aligned}$$

and rearranging, equation (24) gives

$$(2-\gamma\sin^2k)^2\sin^2\ell\cos^2k-\gamma^2\cos^2\ell\sin^2k\cos^2k=0\,,$$

which implies

$$\left[(2-\gamma\sin^2k)\sin\ell\pm\gamma\sin k\cos\ell\right]\cos k=0\,. \quad (25)$$

If we consider $k<\pi/2$, the term of $\cos k\neq 0$. Thus, we have

$$(2-\gamma\sin^2k)\sin\ell\pm\gamma\sin k\cos\ell=0\,.$$

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

$$(2-\sin^2k)\sin\ell\pm\sin k\cos\ell=0\,, \quad (26)$$

which gives

$$\tan\ell=\mp\frac{\sin k}{2-\sin^2k}\,. \quad (27)$$

[1]

[1] 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 (25) agrees with my computer algebra of the next section?? May be best to update Reduce so that the command `in_tex` works.

4 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 $\xi =$ 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) when k
    , ginv(~a(j+~k)^2,t)=>a(j+k+1)^2-ginv(a(j+k+2)^2,t) when k
    , 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;
```

5 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(j
                    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  *gar
                    2               2
```

```

            1                1                1
      *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 + j)
            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  *gam
                    2               2

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

            1                1                1
      *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 + j)
            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  *gam
                     2               2

            1                1                1
      *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 + j)
            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   *gam
                       2                2

                1                1                1
      *( - ----*hh(1 + j) + ----*hh(3 + j) + ----*hh( - 1 + j)
                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   *ga
                         2                2

               1                1                 1
      *( - ----*uu(1 + j) + ----*uu(3 + j) + ----*uu( - 1 + j)
               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 *gam
               6                          120
```

```

            1                4    3     1                 6
         + ----*df(uu,x,5)*dd *gam  - ------*df(uu,x,7)*dd *ga
            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 *gam
            6                         120

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

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