Computer algebra derives discretisations of the stochastically forced Burgers' partial differential equation

A. J. Roberts*

January 29, 2006

Abstract

The computer algebra routine[1] documented here empower you to reproduce and check many of the details described in [10]. We consider a region of a spatial domain far from any boundaries, and apply stochastic centre manifold techniques to derive a discrete model for the dynamics. The approach automatically parametrises the microscale structures induced by spatially varying stochastic noise within each element. The crucial aspect of this work is that we explore how many noise processes and neighbouring elements may interact in nonlinear dynamics.

*Computational Engineering and Science Research Centre, Department of Mathematics & Computing, University of Southern Queensland, Toowoomba, Queensland 4352, Australia. mailto:aroberts@usq.edu.au

[1]The computer algebra routines are written in the package REDUCE. At the time of writing, information about REDUCE was available via http://www.reduce-algebra.com/. Free demonstration versions of REDUCE were available and will execute these routines.

Contents

1 Introduction

Consider the stochastically forced Burgers' equation

$$\frac{\partial u}{\partial t} = -\alpha u\frac{\partial u}{\partial x} + \frac{\partial^2 u}{\partial x^2} + \sigma\phi(x,t)\,, \tag{1}$$

on a domain large enough so the boundaries are immaterial. I seek a discrete model in the interior of the domain. In particular, I model effects quadratic in the noise amplitude σ. We seek a normal form where the model involves no fast-time convolutions. Throughout we adopt the Stratonovich interpretation of stochastic differential equations so that the ordinary rules of calculus apply.

Artificial internal boundary conditions discretise Discretise in spce through placing equi-spaced grid points at x_j a distance h apart. Then the jth element is notionally $|x - x_j| < h/2$. Form the elements by introducing the artificial internal boundary conditions (IBCs)

$$u_j(x_{j\pm1},t) - u_j(x_j,t) = \gamma\left[u_{j\pm1}(x_{j\pm1},t) - u_j(x_j,t)\right]\,, \tag{2}$$

where $u_j(x,t)$ denotes the field within the jth element. The coupling parameter γ controls the flow of information between adjacent elements: when $\gamma = 0$, adjacent elements are decoupled; when $\gamma = 1$, the field in the jth element must extrapolate to the neighbouring elements field at its grid point.

I proved [6] these IBCs ensured the discrete models are consistent with the linear autonomous dynamics to high order in small element size h; all examples also show these IBCs produce high order consistency for the nonlinear autonomous dynamics but no proof yet exists.

Centre manifold theory underlies the modelling Centre manifold theory [1, 2, e.g.] is an incredibly useful basis for modelling [7, 8, e.g.]. Here a useful base for the discrete modelling is the linear, decoupled, no-noise case $\alpha = \gamma = \sigma = 0$ when Burgers' equation (1) becomes the linear diffusion equation on each element:

$$\frac{\partial u_j}{\partial t} = \frac{\partial^2 u_j}{\partial x^2} \quad \text{such that} \quad u_j(x_j \pm h, t) - u_j(x_j, t) = 0\,. \tag{3}$$

The linear modes are then the complete, orthogonal trigonometric functions $\cos(n\theta)\exp(-\beta_n t)$ for n even, and $\sin(n\theta)\exp(-\beta_n t)$ for n odd, for $n = 0, 1, 2, \ldots$ with phase $\theta = \pi(x - x_j)/h$, that changes by π across each element, and decay rate $\beta_n = (\pi n/h)^2$.[2] Thus for each of, say, m elements there is one zero eigenvalue and the the others are negative, $\leq -\beta_1 = -\pi^2/h^2$. There are also 3 zero eigenvalues of the trivial equations for the parameters, $\alpha_t = \gamma_t = \sigma_t = 0$. Hence there exists an $m+3$ dimensional centre eigenspace which under nonlinear, coupling and forcing perturbations becomes an $m+3$ dimensional centre manifold. The centre manifold may be prarametrised by the grid value U_j in each element and the three parameters α, γ and σ. As set up here the centre manifold approximations are global in the grid values $\mathbf{U}$ but local in the three parameters α, γ and σ.

[2] There are also the even wavenumber sine functions $\sin(n\theta)\exp(-\beta_n t)$ for $n = 2, 4, 6, \ldots$. To eliminate these from the complete set and obtain orthogonality over each element, recast the IBCs.

1.1 Model nonlinear dynamics

A low accuracy discrete model, in terms of centred difference δ and centred mean μ operators, is

$$\begin{aligned} \dot{U}_j &= \frac{\gamma}{h^2}(1+\tfrac{1}{12}\alpha^2h^2U_j^2)\delta^2U_j - \alpha\frac{\gamma}{h}U_j\mu\delta U_j \\ &\quad + \sigma\left[\phi_{j,0} - \alpha\frac{2h}{\pi^2}\phi_{j,1}U_j - \alpha^2\frac{8h^2}{3\pi^4}\phi_{j,2}U_j^2\right] + \mathcal{O}\big(\alpha^3,\gamma^2+\sigma^2\big)\,, \quad (4) \end{aligned}$$

when the subgrid scale noise in each element is truncated to

$$\phi(x,t) = \phi_{j,0}(t) + \phi_{j,1}(t)\sin\theta + \phi_{j,2}(t)\cos 2\theta\,.$$

The first line is the so-called holistic discretisation for the deterministic Burgers' equation whose good properties on finite sized elements I have already explored [5]; in particular, see that the nonlinearly enhanced diffusion stabilises the scheme for non-small field u. The second line contains the first approximation to some noise influences: the nonlinearity in Burgers' equation transforms the additive noise of the SPDE (1) into multiplicative noise components in the discretisation. Simple numerical schemes miss the multiplicative noise terms.

The correspondingly low accuracy, subgrid scale structure within each element is

$$\begin{aligned} u_j(x,t) &= U_j + \gamma\left[\tfrac{1}{2}(\theta/\pi)^2\delta^2 + (\theta/\pi)\mu\delta\right]U_j \\ &\quad + \alpha\gamma\tfrac{1}{6}h\left[(\theta/\pi)^3 - (\theta/\pi)\right]U_j\delta^2U_j \\ &\quad + \sigma\left[\sin\theta\, Z_1\phi_{j,1} + \cos 2\theta\, Z_2\phi_{j,2}\right] \\ &\quad + \sigma\alpha 2\left[\frac{h}{\pi^2}Z_1\phi_{j,1} + \frac{8}{3h}\sin\theta\, Z_{1,2}\phi_{j,2} - \frac{2}{3h}\cos 2\theta\, Z_{2,1}\phi_{j,1}\right]U_j \\ &\quad + \mathcal{O}\big(\alpha^2,\gamma^2+\sigma^2\big)\,. \quad (5) \end{aligned}$$

where the operators Z denote the specific convolutions (11) over the immediate past history of the noise component to which they are applied:

1. the first line is classic Lagrangian interpolation through the surrounding grid values;

2. the second line refines the deterministic structure due to the dynamics of the nonlinear advection term, see [5];

3. the third line shows that the components of the instantaneous noise, $\phi_{j,1}$ and $\phi_{j,2}$, in the element affect the subgrid structure through a memory integral, $Z_p\phi_{j,p}$ over the immediate past history of the noise component as the influence of the instantaneous forcing diffuses across the element—the higher wavenumber component having a shorter memory;

4. and lastly, the fourth line shows how the interactions of the noise components with the nonlinear advection, ameliorated by the intra-element diffusion, generate additional subgrid structure in the field u.

Theory empowers systematic extensions The model (4–5) may be extended in many ways to ensure more accuracy:

- compute higher order terms in the coupling parameter γ to obtain discretisations of higher order consistency in the element size h as the method discovers more information about the interaction between elements;

- compute higher order terms in the nonlinearity parameter α to discover more about the subgrid scale interactions within an element;

- compute more of the forcing terms, those in σ and σ^2, to discover more noise interactions;

- increase the number of resolved Fourier modes of the applied noise forcing to resolve more of the forcing;

- and, lastly, increase the number of Fourier modes used to resolve subgrid scale noise structures in the field u within each element.

The Approximation Theorem of centre manifold theory supports any of these truncations—except the last. Rigorous theory only applies after we have projected the dynamics onto a finite number of subgrid Fourier modes.

1.2 Diffusion correlates noise across space

The simple case of stochastically forced diffusion, nonlinear parameter $\alpha = 0$, exhibits that noise processes at different grid points should be correlated in the discretisation even though the original applied noise has no spatial correlations. A low accuracy asymptotic model for the evolution of each grid value is

$$\begin{aligned}\dot{U}_j &= \gamma\frac{1}{h^2}\delta^2 U_j + \sigma\phi_{j,0} \\ &\quad + \gamma\sigma\delta^2\left[-\frac{1}{24}\phi_{j,0} + \frac{1}{4\pi^2}\phi_{j,2} + \frac{1}{16\pi^2}\phi_{j,4} + \frac{1}{36\pi^2}\phi_{j,6} + \cdots\right] \\ &\quad + \mathcal{O}(\alpha, \sigma^3, \gamma^2)\,. \end{aligned} \tag{6}$$

Because simple diffusion is linear, there are no stochastic interaction terms, those of $\mathcal{O}(\sigma^2)$. But note that the noise applied to the jth grid value U_j is coupled to the noise sources in neighbouring elements through the second difference $\delta^2\phi_{j,n}$ terms. This coupling arises because the noise in one element creates spatial structures that diffuse out into neighbouring elements and affect the evolution. The coupling does not depend upon element size because the diffusion time into a neighbouring element has the same time scale as diffusive decay within each element.

For interest, the corresponding subgrid scale field within each element, but only with three Fourier modes, is

$$\begin{aligned} u_j(x,t) &= U_j + \gamma\left[(\theta/\pi)\mu\delta + \tfrac{1}{2}(\theta/\pi)^2\delta^2\right]U_j \\ &\quad + \sigma\left[\sin\theta\, Z_1\phi_{j,1} + \cos\theta\, Z_2\phi_{j,2}\right] \\ &\quad + \gamma\sigma\Big[(\theta/\pi)\mu\delta Z_2\phi_{j,2} + (-\tfrac{1}{24} - \tfrac{1}{4\pi^2} + \tfrac{1}{2}(\theta/\pi)^2)\delta^2 Z_2\phi_{j,2} \\ &\qquad + \sin\theta\,\mu\delta Z_1\left(-\tfrac{4}{\pi^2}\phi_{j,0} - \tfrac{4}{\pi^2}\phi_{j,2} + \tfrac{16}{h^2}Z_2\phi_{j,2}\right) \end{aligned}$$

$$+\cos 2\theta\,\delta^2 Z_2\left(\tfrac{1}{2\pi^2}\phi_{j,0}+\tfrac{1}{2\pi^2}\phi_{j,2}-\tfrac{2}{h^2}Z_2\phi_{j,2}\right)\Big]$$
$$+\mathcal{O}(\alpha,\sigma^3,\gamma^2)\,. \tag{7}$$

The first line is the deterministic model which for linear problems is classic Lagrange interpolation. The second line shows how forced modes decay in each element. The remaining lines indicate how the noise in adjoining elements affect each other as their influence diffuses out.

The asymptotic approximation (6) models forced diffusion dynamics, (1) with $\alpha = 0$, only when we set the coupling parameter $\gamma = 1$. Undesirably, the resultant model

$$\dot{u}_j \approx \frac{1}{h^2}\delta^2 u_j + \sigma\phi_{j,0} + \sigma\delta^2\left[-\frac{1}{24}\phi_{j,0}+\frac{1}{4\pi^2}\phi_{j,2}+\frac{1}{16\pi^2}\phi_{j,4}+\frac{1}{36\pi^2}\phi_{j,6}+\cdots\right], \tag{8}$$

has infinite sums of noise. But these noises are unknown. Thus we may combine the infinite sums of noise terms into new unknown noises with the same statistics as the infinite sums. Let us explore two more different versions.

1. The combination, extrapolated to an infinite number of terms,

$$\frac{1}{4\pi^2}\phi_{j,2}+\frac{1}{16\pi^2}\phi_{j,4}+\frac{1}{36\pi^2}\phi_{j,6}+\cdots = \frac{1}{4\pi^2}\sum_{n=1}^{\infty}\frac{1}{n^2}\phi_{j,2n}$$
$$\equiv \frac{1}{4\pi^2}\sqrt{\sum_{n=1}^{\infty}\frac{1}{n^4}}\,\hat{\phi}_j(t) = \frac{1}{12\sqrt{10}}\,\hat{\phi}_j(t)$$

where the effectively new stochastic noise $\hat{\phi}_j(t)$ represents the cumulative effect of the infinite sum of the stochastic components $\phi_{j,2n}$. Thus the model (8) becomes

$$\dot{u}_j \approx \frac{1}{h^2}\delta^2 u_j + \sigma\left[\phi_{j,0}-\tfrac{1}{24}\delta^2\phi_{j,0}+\tfrac{1}{12\sqrt{10}}\delta^2\hat{\phi}_j\right]. \tag{9}$$

This model has only $2m$ noise modes for a spatial domain with m elements.

2. A further possible simplification of the model combines the two second difference δ^2 terms through replacing the noise components by the orthogonal combination

$$\begin{bmatrix}\psi_j(t)\\ \hat{\psi}_j(t)\end{bmatrix} = \begin{bmatrix}\sqrt{\frac{5}{7}} & -\sqrt{\frac{2}{7}}\\ \sqrt{\frac{2}{7}} & \sqrt{\frac{5}{7}}\end{bmatrix}\begin{bmatrix}\phi_j(t)\\ \hat{\phi}_j(t)\end{bmatrix}.$$

Then the model (9) becomes

$$\dot{u}_j \approx \frac{1}{h^2}\delta^2 u_j + \sigma\left[\sqrt{\tfrac{5}{7}}\psi_j - \tfrac{1}{24}\sqrt{\tfrac{7}{5}}\delta^2\psi_j + \sqrt{\tfrac{2}{7}}\hat{\psi}_j\right]. \tag{10}$$

Although both these models have $2m$ noise modes, I prove [10, Appendix] that we cannot reduce this to the minimal m modes without making the model undesirably nonlocal. Centre manifold theory supports the particular weighted combination of noise in the model (10) as providing an appropriate balance between noise correlated between neighbouring elements, and independent noise in each element. Thus I commend model (10) as a simple three point discretisation for the stochastically forced diffusion equation.

2 Overview the computer algebra algorithm

Parameters Compute the asymptotic series to this order of truncation in the three small parameters α, γ and σ. However, to control flexibly the truncation of the asymptotic expansion we may couple the scaling of the small parameters. For example, put $\sigma = \texttt{sig}$, $\gamma = \texttt{gam}$ and $\alpha = \texttt{alf}\cdot\texttt{gam}$ to compute asymptotic expansions correct to residuals $\mathcal{O}\left(\sigma^p, \gamma^q + \alpha^q\right)$.

```
▷▷ sfbe ◁◁
let { alf^3=>0, sig^3=>0, gam^3=>0 };
sigma:=sig;
gamma:=gam;
alpha:=alf*gam;
```

Table 1: number of terms in the evolution $\dot{U}_j = g_j(\mathbf{U}, t)$ when only three Fourier modes are used: the numbers in *italics* count the terms in the low accuracy nonlinear model (4); whereas the numbers in **bold** count the terms of the low accuracy model (6) of diffusion. Expect many more terms if using more Fourier modes.

	σ^0					σ^1					σ^2				
α^3	0	0			α^3	1	13			α^3	9				
α^2	*0*	*3*	14		α^2	*1*	16	82		α^2	6	156			
α^1	*0*	*2*	8	19	α^1	*1*	11	45	93	α^1	3	42	238		
α^0	***0***	***3***	5	7	α^0	***1***	**6**	10	14	α^0	**0**	**0**	0	0	
	γ^0	γ^1	γ^2	γ^3		γ^0	γ^1	γ^2	γ^3		γ^0	γ^1	γ^2	γ^3	

Within each element, we express the additive forcing noise in the spectral Fourier modes. Truncate the infinite sum to `nfn` modes in each element. For approximations $\mathcal{O}(\sigma^3, \gamma^3 + \alpha^3)$ may compute the asymptotic expansion with four Fourier modes in about one minute CPU time on my current notebook, but five Fourier modes exhausts RAM.

▷▷ sfbe ◁◁+

```
nfn:=3;
```

Stochastically forced modes depend upon time and so we have to solve for the subgrid structure in the spectral Fourier modes. The parameter `nft` is the number of Fourier modes resolved in the subgrid structure. The $\mathcal{O}(\alpha^2, \sigma^2)$ terms in the model seem to depend upon this truncation.

▷▷ sfbe ◁◁+

```
nft:=nfn;
```

Table 1 indicates the level of complexity of the multiparameter asymptotic expansion via a sort of Newton diagram. Note: the table reports the number of terms in the various terms of the model $\dot{U}_j = g_j(\mathbf{U}, t)$, there are vastly more terms describing the subgrid scale structure within each element.

Overview the algorithm First find the deterministic model, then second determine the noise effects. The derivation of the noise effects has differences from the derivation of the deterministic modelling that make this two stage process worthwhile. The need to use spectral Fourier modes for noise modelling seems to be the reason. Finally, assume the noise terms are stochastic and transform the quadratic noises into new effective noises.

▷▷ sfbe ◁◁+

```
% see caddsfbe.pdf for documentation
◁◁ preamble ▷▷
◁◁ deterministic preamble ▷▷
◁◁ deterministic iteration ▷▷
◁◁ noise preamble ▷▷
◁◁ noise iteration ▷▷
◁◁ save results ▷▷
◁◁ centred differences ▷▷
◁◁ transform quadratic noise ▷▷
end;
```

3 Preamble

Trivially improve printing, whenever the parameters are still variables.

▷▷ preamble ◁◁

```
on div; off allfac; on revpri;
if alf neq 0 then factor alf;
if gam neq 0 then factor gam;
if sig neq 0 then factor sig;
factor h;
```

Make subgrid structures a function of element phase $\theta = \pi(x - x_j)/h$ for element size h; denote phase θ by th. Observe: the phase θ changes by π

over one element; the boundaries of the jth element are at $\theta = \pm\pi/2$; and the adjacent grid point are at $\theta = \pm\pi$.

▷▷ preamble ◁◁+

```
depend th,x;
let df(th,x)=>pi/h;
```

Procedure `mean` computes the mean over the jth element: it is used for finding solvability conditions and for computing Fourier transforms.

▷▷ preamble ◁◁+

```
procedure mean(f);
%     int(f,th,-pi/2,pi/2)/pi;
%     meano(trigsimp(f,combine),th);
    (meano(f,th) where means);
```

However, REDUCE is twice as fast using the dedicated transformation rules encoded in operator `meano` when compared to the generic definite integration function.

▷▷ preamble ◁◁+

```
operator meano;
linear meano;
means:={ meano(1,th) => 1
    , meano(th,th) => 0
    , meano(th^~p,th) => (pi/2)^p*(1+(-1)^p)/2/(p+1)
    , meano(cos(~m*th),th) => 2*sin(m*pi/2)/m/pi
    , meano(cos(th),th) => 2/pi
    , meano(sin(~a),th) => 0
    , meano(th*cos(~a),th) => 0
    , meano(th*sin(th),th) => meano(cos(th),th)
    , meano(th*sin(~m*th),th) => (
      -cos(m*pi/2) +meano(cos(m*th),th) )/m
    , meano(th^~p*cos(th),th) =>
```

```
  +(pi/2)^(p-1)*(1+(-1)^p)/2
  -p*meano(th^(p-1)*sin(th),th)
, meano(th^~p*sin(th),th) =>
  +p*meano(th^(p-1)*cos(th),th)
, meano(th^~p*cos(~m*th),th) => (
  +(pi/2)^(p-1)*sin(m*pi/2)*(1+(-1)^p)/2
  -p*meano(th^(p-1)*sin(m*th),th) )/m
, meano(th^~p*sin(~m*th),th) => (
  -(pi/2)^(p-1)*cos(m*pi/2)*(1-(-1)^p)/2
  +p*meano(th^(p-1)*cos(m*th),th) )/m
}$
```

Parametrise solutions by an evolving amplitudes $U_j(t)$ (or 'order parameter'). Its evolution is $dU_j/dt = \dot{U}_j = g_j(\mathbf{U}, t, \sigma)$.

▷▷ preamble ◁◁+

```
operator u;
depend u,t;
let df(u(~k),t)=>sub(j=k,gj);
```

Use this function to trace print the residuals.

▷▷ preamble ◁◁+

```
procedure mylength(res);
    if res=0 then 0 else length(res);
```

4 The deterministic model

The algorithm for the holistic discretisation of the unforced ($\sigma = 0$) Burgers' equation is well established [5, e.g.]. The only difference is the use of phase θ

as the subgrid spatial coordinate within each element rather than $\xi = \theta/\pi = (x - x_j)/h$ of earlier work.

The most basic linear approximation to the dynamics on the element is $u = U_j$ (piecewise constant) where $\dot{U}_j = 0$ (a fixed point).

▷▷ deterministic preamble ◁◁

```
uj:=u(j);
gj:=0;
```

Iterate towards a solution of the deterministic problem. The parameter α, `alpha`, controls the truncation in nonlinearity. The iteration terminates when the residual of the modified Burgers' equation is zero to the specified order of nonlinearity. I [4] explained the centre manifold rationale and the computational effectiveness of this simple algorithm.

▷▷ deterministic iteration ◁◁

```
repeat begin
    res:=-df(uj,t)+df(uj,x,2)-alpha*uj*df(uj,x);
    bcr:=(sub(th=+pi,uj)-u(j))-gamma*(u(j+1)-u(j));
    bcl:=(sub(th=-pi,uj)-u(j))-gamma*(u(j-1)-u(j));
    gj:=gj+(gd:=mean(res)-(bcr+bcl)/h^2);
    uj:=uj+solv(res-gd,xt)+th*(-bcr+bcl)/2/pi;
    showtime;
end until res=0 and bcr=0 and bcl=0;
```

The linear operator `solv` used above solves $\mathcal{L}v = v_t - \frac{\pi^2}{h^2} v_{\theta\theta} = \text{RHS}$ such that $v(0,t) = 0$ and $v(\pi,t) = v(-\pi,t)$. We introduce the variable `xt` to catch spatial variations, through θ, and fast time variations in the noise terms.

▷▷ deterministic preamble ◁◁+

```
operator solv;
linear solv;
depend th,xt;
let { solv(th^~p,xt) => (h/pi)^2*( -th^(p+2)
```

```
      +th*pi^(p+1)*(1-(-1)^p)/2 )/(p+2)/(p+1)
, solv(th,xt) => (h/pi)^2*(-th^3+th*pi^2)/6
, solv(1,xt) => (h/pi)^2*(-th^2)/2
}$
```

5 Linear noise effects

Introduce the noise in the jth element in its spatial Fourier decomposition

$$\phi(x,t) = \sum_{n=0}^{\infty} \phi_{j,n}(t) \times \begin{cases} \cos n\theta\,, & n \text{ even}, \\ \sin n\theta\,, & n \text{ odd}, \end{cases}$$

where the trigonometric dependence is in multiples of the phase $\theta = \pi(x - x_j)/h$. These subgrid scale modes, $\sin n\theta$ and $\cos n\theta$, are the spectral modes of the diffusion operator on each element and so are the natural modes for describing much of the subgrid scale dynamics. However, truncate the spatial structure of noise to a finite number of components, namely `nfn`.

```
▷▷ noise preamble ◁◁
operator phi;
noise:=for n:=0:nfn-1 sum
    phi(j,n)*(if evenp(n) then cos(n*th) else sin(n*th));
```

Linearise products of trigonometric functions, although `trigsimp` seems about 20% faster.

```
▷▷ noise preamble ◁◁+
let { sin(~a)*cos(~b) => (sin(a+b)+sin(a-b))/2
    , cos(~a)*cos(~b) => (cos(a-b)+cos(a+b))/2
    , sin(~a)*sin(~b) => (cos(a-b)-cos(a+b))/2
    , cos(~a)^2       => (1+cos(2*a))/2
    , sin(~a)^2       => (1-cos(2*a))/2
```

```
};
```

Let `tt` label the fast time of stochastic fluctuations so we can separate the fast stochastic fluctuations from the slow evolution of the amplitudes u_j. Also introduce `xt` to label both the subgrid (fast) spatial scales and the fast time scales so we can group all factors in the fast space-time dynamics.

```
                    ▷▷ noise preamble ◁◁+
depend tt,t;
depend phi,tt,xt;
```

Define β_n to be the decay rate of linear modes on the element, here $\beta_n = \pi^2 n^2/h^2$, $n = 0, 1, 2, \ldots$, so that the spatial modes decay linearly like $\cos|\sin(nx)\exp(-\beta_n t)$.

```
                    ▷▷ noise preamble ◁◁+
procedure beta(n); pi^2*n^2/h^2;
```

5.1 Memory convolutions

Now let $Z_{\mathbf{p}}$ denote the operator of convolutions in time of any term, that is, the operator

$$Z_{\{p_1,p_2,\ldots\}} = \exp(-\beta_{p_1} t) \star \exp(-\beta_{p_2} t) \star \cdots \star \quad \text{and} \quad Z_{\{\}} = 1\,; \tag{11}$$

so

$$\partial_t Z_{\{p_1,p_2,\ldots\}} = -\beta_{p_1} Z_{\{p_1,p_2,\ldots\}} + Z_{\{p_2,\ldots\}}\,. \tag{12}$$

Note: the order of the convolutions does not matter as I have already established [9, Appendix], but it seems crucial for the residual computations to terminate. Further, each such convolution may be written as an Ito integral of an integrand that only depends upon the factors β_{p_k}.

```
                ▷▷ noise preamble ◁◁+
operator z;
depend z,tt,xt;
let { z({},~a)=>a
    , z(~p,z(~q,~a)) => z(append(p,q),a)
    , df(z(~p,~a),t) => df(z(p,a),tt)
    , df(z(~p,~a),tt)
      => -beta(first p)*z(p,a)+z(rest p,a)
    };
```

5.2 Iterate to the asymptotic solution

Iterate to derive effects in the additive noise. This iteration starts from the deterministic model already deduced. Finish the iteration when all residuals are zero.

```
                ▷▷ noise iteration ◁◁
it:=1$
repeat begin
◁◁ ibcs ▷▷
◁◁ spdes ▷▷
showtime;
end until res=0 and bcr=0 and bcl=0 or(it:=it+1)>19;
```

5.3 Compute residuals

Compute residual of the Burgers' SPDE (1).

```
                ▷▷ spdes ◁◁
res:=-df(uj,t)+df(uj,x,2)-alpha*uj*df(uj,x)+sigma*noise;
```

Linearise products of trigonometric functions, then use Fourier series to express the spatial structure associated with any noise terms, truncating to the parameter nft.

```
                    ▷▷ spdes ◁◁+
res:=fourier(res);
write length_res:=mylength(res);
```

We only find Fourier transforms of components in the noise.

```
                ▷▷ noise preamble ◁◁+
procedure fourier(a);
    coeffn(a,sig,0)+(ft(a-coeffn(a,sig,0),th) where ffts);
```

Note: the overlap in the pattern matching below should not be a problem as REDUCE should meet the trigonometric cases first and deal with them; thus the full Fourier series machinations will only be done for terms involving polynomials in phase θ. The main effect of this truncation in Fourier modes seems at first sight to be in σ^2 terms.

```
                ▷▷ noise preamble ◁◁+
operator ft;
linear ft;
ffts:={ ft(1,th)=>1
    , ft(cos(~m*th),th) => cos(m*th) when evenp(m)
    , ft(sin(~m*th),th) => sin(m*th) when not evenp(m)
    , ft(~a,th) => mean(a)+for m:=1:nft-1 sum 2*(
        if evenp(m) then cos(m*th)*mean(cos(m*th)*a)
                    else sin(m*th)*mean(sin(m*th)*a))
    }$
```

Because we have to abandon strict definition of the grid values U_j to avoid memory integrals, we revert to the general form of the coupling conditions.

▷▷ ibcs ◁◁

```
uj0:=sub(th=0,uj);
bcr:=(sub(th=+pi,uj)-uj0)-gamma*(sub(j=j+1,uj0)-uj0);
write length_bcr:=mylength(bcr);
bcl:=(sub(th=-pi,uj)-uj0)-gamma*(sub(j=j-1,uj0)-uj0);
write length_bcl:=mylength(bcl);
```

5.4 The residuals update the asymptotic approximation

First, update the field in the jth element from the residuals of the inter-element coupling conditions.

▷▷ ibcs ◁◁+

```
uj:=uj+th*(-bcr+bcl)/(2*pi)-th^2*(bcr+bcl)/pi^2/2;
```

Update the evolution of the jth "grid value". Recall the equation for updates u' and g' is $g'+\mathcal{L}u'=$ residual, where now the operator $\mathcal{L}=\partial_t-\partial_{xx}$ includes fast time variations. The operator `nonint` extracts from the residual all those terms which would generate generate secular growth in the field u and so instead must be placed in the model's evolution g. The last rule here comes from integration by parts and is essential in order to eliminate memory integrals (convolutions) in the model evolution.

▷▷ noise preamble ◁◁+

```
operator nonint;
linear nonint;
nonints:={ nonint(1,tt)=>1
    , nonint(phi(~j,~n),tt) => phi(j,n)
    , nonint(z(~p,~a),tt)
      => nonint(a,tt)/(for each m in p product beta(m))
    }$
```

Simplest to update the evolution direct from the residual of the SPDE after the IBCs have corrected the field u_j.

```
▷▷ spdes ◁◁+
gj:=gj+(gd:=(nonint(mean(res),tt) where nonints));
```

Extend the inverse operator `solv` to terms with fast time variations as well as fast (subgrid) space variations.

Note: with the applied forcing, stochastic or deterministic, we have to abandon the strict requirement that U_j be the grid value so that we can eliminate the memory integrals. We could subtract one from the following cosine rule to attempt to maintain U_j as the grid value as far as possible, but it does not work. The requirement for no memory convolutions in `gj` forces us to abandon the definition of U_j as the strict grid values.

```
▷▷ noise preamble ◁◁+
let { solv(z(~p,~a),xt) => (solv(z(rest p,a),xt)-z(p,a))
                              /beta(first p) when p neq {}
    , solv(~a*cos(~m*th),xt) => z({m},a)*cos(m*th)
      when df(a,th)=0
    , solv(~a*sin(~m*th),xt) => z({m},a)*sin(m*th)
      when df(a,th)=0
    , solv(~a*sin(th),xt) => z({1},a)*sin(th)
      when df(a,th)=0
    }$
```

Lastly, update the field from the residual of the SPDE.

```
▷▷ spdes ◁◁+
uj:=uj+solv(res-gd,xt);
```

6 Integrate quadratic memory convolutions

To extract quadratic corrections to the evolution, use integration by parts so all non-integrable convolutions are reduced to the canonical form of the convolution being entirely over one noise in a quadratic term, namely $\phi_{j,n} Z_{\mathbf{p}} \phi_{k,m}$. Rewrite the convolution ODE (12) as

$$\beta_p Z_{p.\mathbf{p}'} = -\frac{\partial Z_{p.\mathbf{p}'}}{\partial t} + Z_{\mathbf{p}'} ,$$

where the vector of convolution parameters are $\mathbf{p} = p.\mathbf{p}'$ so that $p = \text{first}\,\mathbf{p}$ and $\mathbf{p}' = \text{rest}\,\mathbf{p}$. Then

$$\begin{aligned}
&\int Z_{p.\mathbf{p}'} Z_{q.\mathbf{q}'}\, dt \\
&= \frac{1}{\beta_p + \beta_q} \int \beta_p Z_{p.\mathbf{p}'} Z_{q.\mathbf{q}'} + Z_{p.\mathbf{p}'} \beta_q Z_{q.\mathbf{q}'}\, dt \\
&= \frac{1}{\beta_p + \beta_q} \int \left[-\frac{\partial Z_{p.\mathbf{p}'}}{\partial t} + Z_{\mathbf{p}'} \right] Z_{q.\mathbf{q}'} + Z_{p.\mathbf{p}'} \left[-\frac{\partial Z_{q.\mathbf{q}'}}{\partial t} + Z_{\mathbf{q}'} \right] dt \\
&= \frac{1}{\beta_p + \beta_q} \int -\frac{\partial}{\partial t} \left[Z_{p.\mathbf{p}'} Z_{q.\mathbf{q}'} \right] + Z_{\mathbf{p}'} Z_{q.\mathbf{q}'} + Z_{p.\mathbf{p}'} Z_{\mathbf{q}'}\, dt \\
&= -\frac{1}{\beta_p + \beta_q} Z_{p.\mathbf{p}'} Z_{q.\mathbf{q}'} + \frac{1}{\beta_p + \beta_q} \int Z_{\mathbf{p}'} Z_{q.\mathbf{q}'} + Z_{p.\mathbf{p}'} Z_{\mathbf{q}'}\, dt .
\end{aligned}$$

Thus integration in time t can be done until we reach terms of the form $\phi_{j,n} Z_{\mathbf{p}} \phi_{k,m}$. These terms must thus go into the evolution equation. Similarly,

$$\int Z^2_{p.\mathbf{p}'}\, dt = -\frac{1}{2\beta_p} Z^2_{p.\mathbf{p}'} + \frac{1}{\beta_p} \int Z_{\mathbf{p}'} Z_{p.\mathbf{p}'}\, dt .$$

▷▷ noise preamble ◁◁+

```
nonints:=append(nonints,{
      nonint(z(~p,~a)*z(~q,~b),tt)
      => nonint(z(rest p,a)*z(q,b)+z(p,a)*z(rest q,b),tt)
      /(beta(first p)+beta(first q))
      when p neq {} and q neq {}
```

```
, nonint(z(~p,~a)^2,tt)
  => nonint(z(rest p,a)*z(p,a),tt)
  /beta(first p) when p neq {}
, nonint(phi(~j,~n)*z(~p,~a),tt) => phi(j,n)*z(p,a)
})$
```

Extend inverse operator `solv` to handle quadratic noise terms that are constant across the element. Use the integrals above.

▷▷ noise preamble ◁◁+

```
let { solv(z(~p,~a)*z(~q,~b),xt) => ( -z(p,a)*z(q,b)
  +solv(z(rest p,a)*z(q,b)+z(p,a)*z(rest q,b),xt) )
  /(beta(first p)+beta(first q))
  when p neq {} and q neq {}
, solv(z(~p,~a)^2,xt)
  => ( -z(p,a)^2/2 +solv(z(rest p,a)*z(p,a),xt) )
  /beta(first p) when p neq {}
}$
```

7 Save the results

First, restore meaning of parameters for ease of later analysis. May later need to include similar code for different truncations of the asymptotic expnasions.

▷▷ save results ◁◁

```
if alpha=alf*gam then begin
    clear alpha;
    gj:=sub(alf=alpha/gam,gj);
    uj:=sub(alf=alpha/gam,uj);
    alpha:=alf;
```

```
end;
```

Output the resulting asymptotic expansions to a file for any further post-processing.

```
▷▷ save results ◁◁+
off nat;
out sfbe_o;
    write gj:=gj;
    write uj:=uj;
    write "end;";
shut sfbe_o;
on nat;
```

8 Maybe rewrite with centred differences

Rewriting in terms of centred difference operators δ and μ is a useful simplification in appearance for at least linear SPDEs: `dd` denotes δ^2; `md` denotes $\mu\delta$.

```
▷▷ centred differences ◁◁
operator dd;
linear dd;
operator md;
linear md;
```

Firstly ensure `md` floats to the front. Secondly, transform $\mu^2 = 1 + \delta^2/4$ [3, p.65, e.g.].

```
▷▷ centred differences ◁◁+
let { dd(md(~a,~j),~i)=>md(dd(a,i),j)
    , md(md(~a,~j),j)=>dd(a,j)+dd(dd(a,j),j)/4
```

```
    };
```

Store expressions for the shift left and right operators: $u_{j\pm k} = (1 \pm \mu\delta + \delta^2/2)^k u_j$ [3, p.65, e.g.]. The variable `dum` is a dummy variable.

▷▷ centred differences ◁◁+

```
array jp(4),jm(4);
depend dum,j;
jp(0):=jm(0):=dum$
jp(1):=dum+md(dum,j)+dd(dum,j)/2$
jm(1):=dum-md(dum,j)+dd(dum,j)/2$
for k:=2:4 do begin
    jp(k):=sub(dum=jp(1),jp(k-1));
    jm(k):=sub(dum=jm(1),jm(k-1));
    end;
```

Invoke the shift operators with this set of rules.

▷▷ centred differences ◁◁+

```
diffs:=
    { u(j+~k) => sub(dum=u(j),jp(+k)) when k>0
    , u(j+~k) => sub(dum=u(j),jm(-k)) when k<0
    , phi(j+~k,~n) => sub(dum=phi(j,n),jp(+k)) when k>0
    , phi(j+~k,~n) => sub(dum=phi(j,n),jm(-k)) when k<0
    }$
```

But we have problems distributing the convolution operator `z` over resultant sums of noise. Thus define a new *linear* operator to do this distribution, adjoin its use to the above rules.

▷▷ centred differences ◁◁+

```
operator zzz;
linear zzz;
```

```
diffs:=(z(~p,~a)=>zzz(a,tt,p)).diffs$
```

Apply the transformation to obtain difference expressions, reverting back to the operator z.

```
▷▷ centred differences ◁◁+
gd:=((gj where diffs) where zzz(~a,tt,~p)=>z(p,a))$
ud:=((uj where diffs) where zzz(~a,tt,~p)=>z(p,a))$
```

9 Transform quadratic noise

Now proceed to transform the strong model to a weak model by replacing the quadratic noises by their effective long term drift and volatility. I previously determined [9] the rationale for the details of this transformation.

9.1 Sort the convolutions

The order of convolutions is immaterial, so sort them all. First, define a simple sorting routing.

```
▷▷ transform quadratic noise ◁◁
procedure place(a,p);
    if p={} then {a} else
    if a<=first p then a.p else
    (first p).place(a,rest p);
procedure sortlist(p);
    if length(p)<2 then p else
    place(first p,sortlist(rest p));
```

Then use zz as a holding operator for z while sorting the convolutions.

```
          ▷▷ transform quadratic noise ◁◁+
operator zz;
gj:=(gj where z(~p,~a)=>zz(sortlist(p),a))$
gj:=(gj where zz(~p,~a)=>z(p,a));
```

9.2 The effectively new noises

Transform the quadratic noise into new noises ψ (`psi`) that are equivalent in their long time statistics. Also rewrite the original noise components in terms of ψ with two subscripts.

```
          ▷▷ transform quadratic noise ◁◁+
operator psi;
depend psi,tt,xt;
```

The operator `long` implements the long-time equivalent noises [9]. For now only transform up to two convolutions. These new noises have subscripts that uniquely identify them.

```
          ▷▷ transform quadratic noise ◁◁+
operator long;
linear long;
let { long(1,tt)=>1
    , long(phi(~j,~n),tt)=>phi(j,n)
    , long(phi(~i,~n)*z({~k},phi(~j,~m)),tt)
      => (if phi(i,n)=phi(j,m) then 1/2 else 0)
      +psi(i,n,j,m,{k})/sqrt(2*beta(k))
    , long(phi(~i,~n)*z({~k2,~k1},phi(~j,~m)),tt)
      => (psi(i,n,j,m,{k1})/sqrt(2*beta(k1))
      +psi(i,n,j,m,{k2,k1})/sqrt(2*beta(k2)))
      /(beta(k1)+beta(k2))
    , abs(h)=>h
```

```
    };
gg:=long(gj,tt)$
```

Invoke the shift operators by adding this set of rules. Note: use variable `i` to denote difference operators in the second variable of the noise process.

```
                ▷▷ centred differences ◁◁+
diffs:=append(diffs,
    { psi(j+~k,~n,~jj,~m,~p)
      => sub(dum=psi(j,n,jj,m,p),jp(+k)) when k>0
    , psi(j+~k,~n,~jj,~m,~p)
      => sub(dum=psi(j,n,jj,m,p),jm(-k)) when k<0
    , psi(~j,~n,j,~m,~p) => psi(j,n,i,m,p)
    , psi(~jj,~n,j+~k,~m,~p)
      => sub({j=i,dum=psi(jj,n,i,m,p)},jp(+k)) when k>0
    , psi(~jj,~n,j+~k,~m,~p)
      => sub({j=i,dum=psi(jj,n,i,m,p)},jm(-k)) when k<0
    })$
```

Now rewrite the long term equivalent model in terms of centred differences. This may be about 2/3 of the length of gg.

```
            ▷▷ transform quadratic noise ◁◁+
gh:=(gg where diffs);
```

9.3 Consolidate the new noises

Create storage for different combinations of noises.

```
            ▷▷ transform quadratic noise ◁◁+
array b(8);
on rounded;
```

```
print_precision 4;
```

Extract all the quadratic noises. I cannot think of a systematic way of doing this other than by my inspection of what arises in the difference form gh.

```
▷▷ transform quadratic noise ◁◁+
gh2:=coeffn(gh,sig,2)$
ga:=coeffn(gh2,alf,1)/h^2$
b(1):=coeffn(ga,gam,0);
```

The α^2 term does not have any differences due to the asymptotic order of truncation.

```
▷▷ transform quadratic noise ◁◁+
b(2):=coeffn(coeffn(gh2,alf,2),h,3)/u(j);
```

Extract the complicated combinations: since there is either none or one of the four difference symbols, I use a binary expansion in the exponent of o.

```
▷▷ transform quadratic noise ◁◁+
ga:=coeffn(ga,gam,1)$
gb:=(ga where { md(~a,i)=>a*o^1, md(~a,j)=>a*o^2,
                dd(~a,i)=>a*o^4, dd(~a,j)=>a*o^8})$
nb:=2$
for each p in coeff(gb,o) do
    if p neq 0 then begin
    nb:=nb+1;
    write b(nb):=p;
    end;
```

Out of interest find the length of each "vector" of noises using an inner product function.

```
▷▷ transform quadratic noise ◁◁+
operator inner;
linear inner;
let { inner(psi(~j,~n,~k,~m,~p),tt) => 0
    , inner(psi(~j,~n,~k,~m,~p)^2,tt) => 1
    , inner(psi(~j,~n,~k,~m,~p)*psi(~u,~v,~w,~x,~y),tt)=>0
    };
array lenb(nb);
for n:=1:nb do write lenb(n):=sqrt(inner(b(n)^2,tt));
```

Now Gramm–Schmidt orthonormalise these vectors. The coefficients in the transform are the coefficients of equivalent noises. I could perhaps improve with some (partial) pivotting, but numerical stability does not yet seem to be an issue.

```
▷▷ transform quadratic noise ◁◁+
array bb(nb,nb);
for n:=1:nb do begin
    for m:=1:n-1 do begin
        write bb(n,m):=inner(b(n)*b(m),tt);
        b(n):=b(n)-bb(n,m)*b(m);
    end;
    write bb(n,n):=sqrt(inner(b(n)^2,tt));
    b(n):=b(n)/bb(n,n);
end;
```

The above ends with a set of orthonormal vectors in `b`. The row operations to transform the original `b` to the orthonormal `b` are

$$\begin{bmatrix} 1 & 0 \\ -c & 1 \end{bmatrix} \quad \text{with inverse} \quad \begin{bmatrix} 1 & 0 \\ c & 1 \end{bmatrix};$$

thus the coefficients form the lower triangular matrix in a QR decomposition of the transpose of the original `b`. Since the orthonormal `b` then are

independent noise sources, we replace the original combinations of noise by those of the lower triangular matrix.

References

[1] J. Carr. *Applications of centre manifold theory*, volume 35 of *Applied Math. Sci.* Springer–Verlag, 1981. 4

[2] Y. A. Kuznetsov. *Elements of applied bifurcation theory*, volume 112 of *Applied Mathematical Sciences*. Springer–Verlag, 1995. 4

[3] National Physical Laboratory. *Modern Computing Methods*, volume 16 of *Notes on Applied Science*. Her Majesty's Stationary Office, 1961. 23, 24

[4] A. J. Roberts. Low-dimensional modelling of dynamics via computer algebra. *Computer Phys. Comm.*, 100:215–230, 1997. 14

[5] A. J. Roberts. Holistic discretisation ensures fidelity to Burgers' equation. *Applied Numerical Modelling*, 37:371–396, 2001. 5, 6, 13

[6] A. J. Roberts. A holistic finite difference approach models linear dynamics consistently. *Mathematics of Computation*, 72:247–262, 2002. http://www.ams.org/mcom/2003-72-241/S0025-5718-02-01448-5. 4

[7] A. J. Roberts. Low-dimensional modelling of dynamical systems applied to some dissipative fluid mechanics. In Rowena Ball and Nail Akhmediev, editors, *Nonlinear dynamics from lasers to butterflies*, volume 1 of *Lecture Notes in Complex Systems*, chapter 7, pages 257–313. World Scientific, 2003. 4

[8] A. J. Roberts. Create useful low-dimensional models of complex dynamical systems. Technical report, http://www.sci.usq.edu.au/staff/robertsa/Modelling/, 2005. 4

[9] A. J. Roberts. Resolve the multitude of microscale interactions to model stochastic partial differential equations. Technical report, http://arxiv.org/abs/math.DS/0506533, June 2005. Submitted to LMS Journal of Computation and Mathematics. 16, 25, 26

[10] A. J. Roberts. Resolve the multitude of microscale interactions to holistically discretise the stochastically forced Burgers' partial differential equation. Technical report, http://arxiv.org/abs/math.DS/0601702, January 2006. 1, 9