

Kuramoto-Sivashinsky Equations

The Kuramoto-Sivashinsky (KS) equation is a nonlinear PDE which arises as a description of flame front flutter of gas burning in a cylindrically symmetric burner. We will consider Kuramoto-Sivashinsky system in the following form,

$$u_t = -\nu u_{xxxx} - u_{xx} + 2uu_x$$

where $(t, x) \in [0, \infty) \times (0, L)$ in periodic domain, $u(t, x) = u(t, x + L)$, and we restrict our solution to the subspace of odd solutions $u(t, -x) = -u(t, x)$.

To develop the Galerkin projection, we expand a periodic solution $u(x, t)$ using a discrete spatial Fourier series,

$$u(x, t) = \sum_{-\infty}^{\infty} b_k(t) e^{ikqx}$$

where $q = \frac{2\pi}{L}$. For simplicity, consider $L = 2\pi$, then $q = 1$. Then, by substituting, we produce the infinitely many evolution equations for the Fourier coefficient,

$$\dot{b}_k = (k^2 - \nu k^4) b_k + ik \sum_{m=-M}^M b_m b_{k-m}$$

where $2M + 1$ is the selected finite number of modes. The coefficients b_k are complex functions of time t . However, by symmetry, we can reduce to a subspace by considering the special symmetry case that b_k is pure imaginary, $b_k = ia_k$ and $a_k \in \mathbb{R}$. Then,

$$\dot{a}_k = (k^2 - \nu k^4) a_k - k \sum_{m=-M}^M a_m a_{k-m}$$

and we can take $N_m = 2M$, and the k can take the values $k = 1, 2, \dots, N_m$ representing N_m -dimensional ODE, which is implemented in function file KSEode.m.

In order to solve the KSE ode for a finite number of modes, we first define:

```
clearvars; close all; clc;

K = 16; %Number of modes N_m
a0 = 0.05*rand(K,1); %Initial condition (small value)

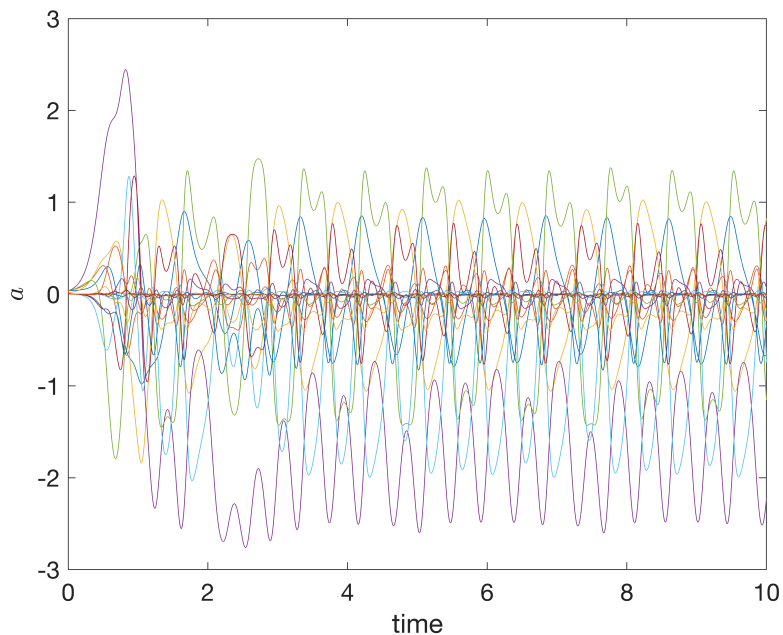
t0 = 0; tfinal = 10; %Initial and final time
h = 0.001; %time step

% Set the ode solver options
options = odeset('RelTol',1e-10,'AbsTol',1e-10);
```

```
% Use ode solver to solve the ode defined
% in the function KSEode.m
[t, a] = ode45(@KSEode,(t0:h:tfinal),a0,options);
```

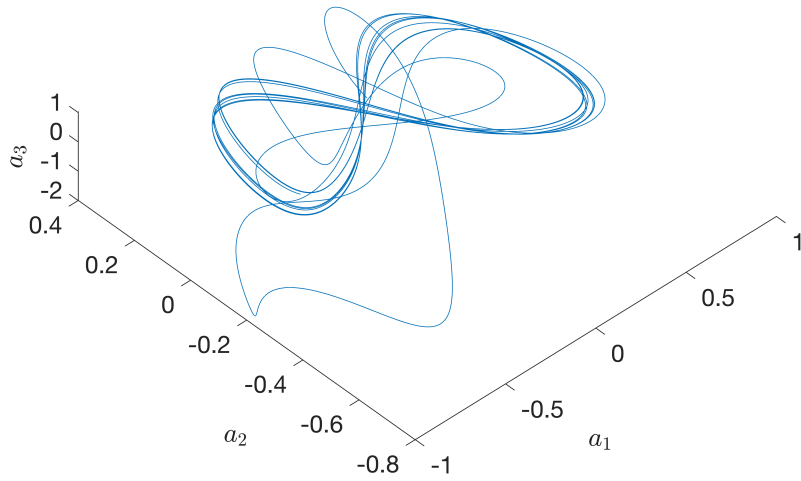
Now, Fourier coefficients is stored in the matrix a . To visualize the data:

```
figure
plot(t,a)
xlabel('time')
ylabel('$a$', "Interpreter", "latex")
set(gca, 'FontSize', 15)
```



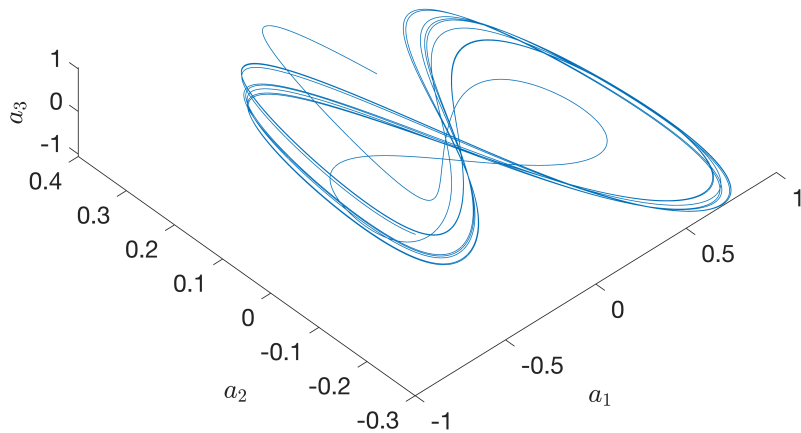
For a more clear view of the dynamic, we can plot sequential modes as a 3D plot. For example, to plot the first three modes we can write:

```
figure
plot3(a(:,1),a(:,2),a(:,3))
view([-43.02 74.81])
xlabel("$a_1$", "Interpreter", "latex")
ylabel("$a_2$", "Interpreter", "latex")
zlabel("$a_3$", "Interpreter", "latex")
set(gca, 'FontSize', 15)
```



Note that you can skip the transient part of the trajectories for more clear view:

```
figure
% plot the last 80% of points
N = round(0.8*length(t));
plot3(a(end-N:end,1),a(end-N:end,2),a(end-N:end,3))
view([-43.02 74.81])
xlabel("$a_1$","Interpreter","latex")
ylabel("$a_2$","Interpreter","latex")
zlabel("$a_3$","Interpreter","latex")
set(gca,'FontSize',15)
```



Now, we can get the time domain solution from the equation (see main text):

$$u(t, x) = \sum_{-\infty}^{\infty} b_k(t) e^{ikqx}$$

as follows:

```
u = zeros(length(t),K); %Initialize time domain solution
x = linspace(0,2*pi,K); % partition spatial domain for K measuring points
for n = 1:length(t) %loop over all trajectories
    for k = 1:length(x) % loop through each spatial point
        u(n,k) = 1.i .* sum(a(n,:).*exp(1.i .* k .* x));
    end
end
```

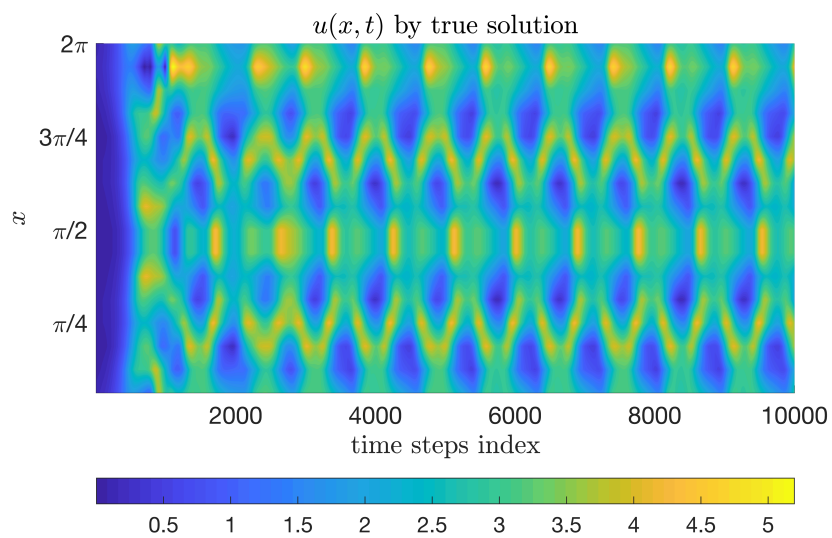
To show the time domain solution as surface plot:

```
figure
h = surf(abs(u)');
pbaspect([2 1 1])
colorbar('southoutside')
ylim([1 size(a,2)])
xlim([1 length(t)])
shading interp

set(gca,'ytick',[0 4 8 12 16])
set(gca,'yticklabels',{'0','\pi/4','\pi/2','3\pi/4','2\pi'})

ylabel('$$$x$$$', 'Interpreter', 'latex')
xlabel('time steps index', 'Interpreter', 'latex')
title('$$$u(x,t)$$$ by true solution', 'Interpreter', 'latex', 'FontSize', 30)
set(gca, 'FontSize', 15)

view([0 90])
```



Now, we can simply save the data to a mat file as follows:

```
FileName = 'KSE_Data';
save(FileName, 'a0', 'a', 'u') % you can add any variable you wish
                               % to save the same way: 'variableName'
```