220311/205001 – Computational engineering

Assignment 4 : Turbulence Burger equation

Boulogne Quentin

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa

SOMMAIRE

1 INTRODUCTION....2
2 RESOLUTION....2
2.1 Methotodolgy to obtain uk....2
2.2 Methotodolgy to obtain Ek....4
3 CODE STRUCTURE....5
4 CODE VALIDATION....6
5 RESULTS DISCUSSION....7
6 CONCLUSION....8
7 ANNEX....8
7.1 Code of Burger equation....8
7.2 Code calculation of Burger equation....9

1 INTRODUCTION

The main objective of this assignment is to gain knowledge about some basic aspects of turbulence such as the energy cascade, the interscale interaction and the roles of the convective and diffusive terms.To do that we will need to solve the Burger's equation for Re=40 with N=20 and N=100.

2 RESOLUTION

2.1 Methotodolgy to obtain uk

$$\begin{aligned} \frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} &= \frac{1}{Re} \Delta \boldsymbol{u} - \nabla p \\ \nabla \cdot \boldsymbol{u} &= 0 \end{aligned}$$

$$Re = \frac{\rho V_0 L}{\mu}$$ The reynolds Number in our case Re = 40 .

$$\Delta t < C_1 \frac{Re}{N^2}$$ The CFL condition is used to limitate the time step used we have Re = 40 ; N= 20 ou 100 ; C1=0.01 to insure convergence.

$$\frac{\partial \vec{u}}{\partial t} + \underbrace{(\vec{u} \cdot \nabla)\vec{u}}_{\text{nonlinear guy in NS}} = \nu \nabla^2 \vec{u} - \nabla p\,; \qquad \nabla \cdot \vec{u} = 0$$

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2} + f$$

By using the Fourrier transformation $u(x) \equiv \sum_{k=-N}^{k=+N} \hat{u}_k e^{ikx}$ we obtain :

$$\frac{\partial \hat{u}_k}{\partial t} + \underbrace{\sum_{p+q=k} \hat{u}_p\, i q\, \hat{u}_q}_{\text{nonlinear guy in Burgers'}} = -\nu k^2 \hat{u}_k + \hat{f}_k$$

The term of forces fk are egal to zero. Then we calculate :

$\sum_{p+q=k} \hat{u}_p\, i q\, \hat{u}_q$ the convective term we call convvection.

νk^2 the diffusion term were ν is the effective one to do the LES calculations.

The discretization equation to obtain uk(n+1) (a complex value).

$$\frac{u_k^{n+1} - u_k^n}{\Delta t} = -(convection + diffusion)$$

$$u_k^{n+1} = u_k^n - \Delta t(convection + diffusion)$$

Then we need to calculate the diffusive term before that.

To calculate the uk in the Fourrier space we need to calculate $\nu_t(k/k_N)$

$$\nu_t(k/k_N) = \nu_t^{+\infty} \left(\frac{E_{k_N}}{k_N}\right)^{1/2} \nu_t^* \left(\frac{k}{k_N}\right)$$

For that we need to calculate those two coefficients:

$$\nu_t^* \left(\frac{k}{k_N}\right) = 1 + 34.5e^{-3.03(k_N/k)} \qquad \nu_t^{+\infty} = 0.31\frac{5-m}{m+1}\sqrt{3-m}C_K^{-3/2}$$

Then we can obtain the effective coefficient

$$\nu_{eff}(k) = \nu + \nu_t(k).$$

The turbulent viscosity term is the term needed to do the LES model. The first curve we draw is for a not LES model in this case we will have :

$$\nu_{eff}(k) = \nu$$

In our case:
m=2
Ck≈ 0.4523 or Ck≈ 0.05
kN=N
EkN is the energy at the cutoff frequency

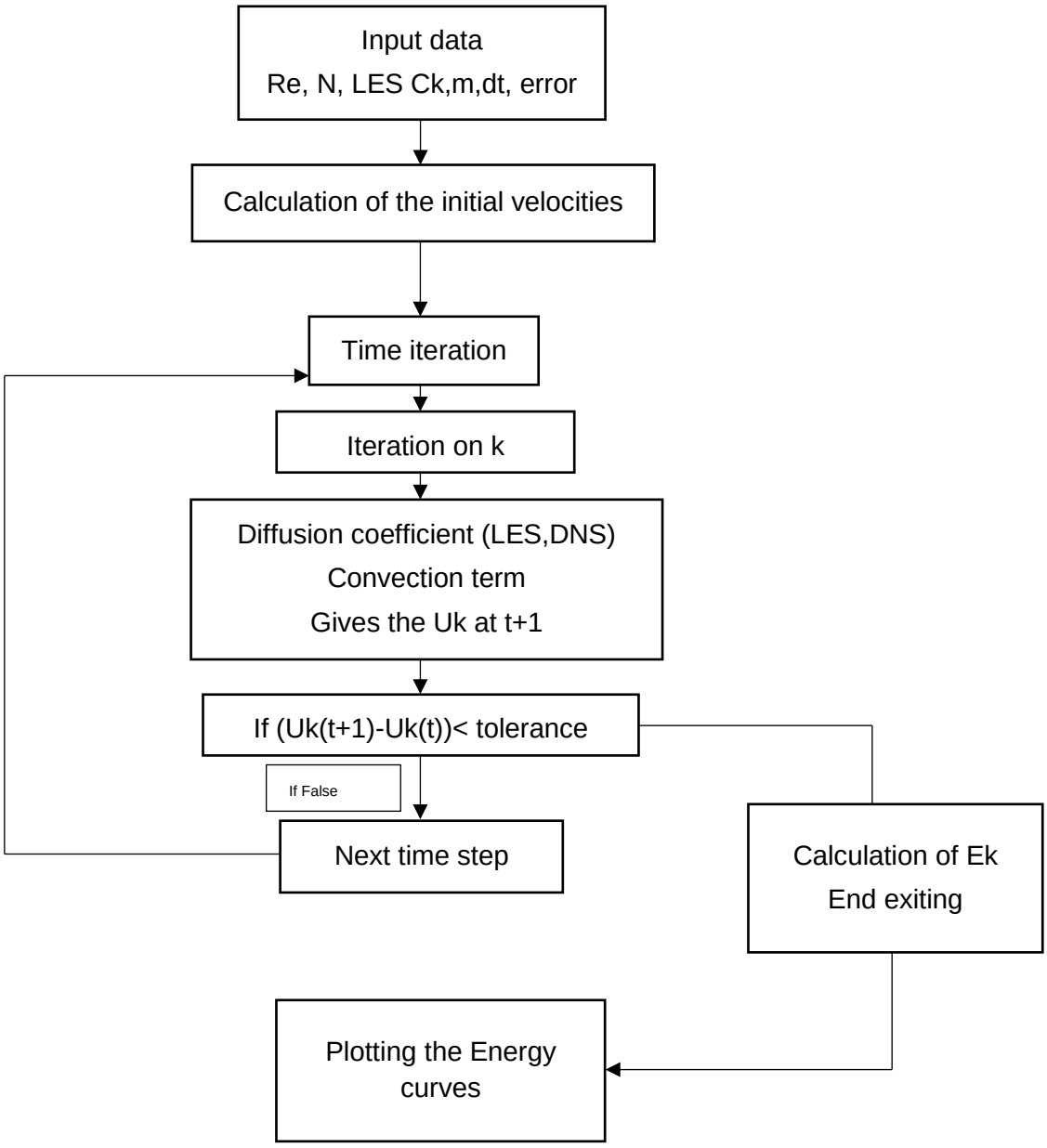
2.2 **Methotodolgy to obtain Ek**

So we know how to calculate the different uk then to calculate Ek we need to iterate on time. We calculate the Ek value by doing:

Ek=uk.conjugate(uk)

3 CODE STRUCTURE

We can observe the code structure :



4 CODE VALIDATION

To validate the code we will look at the curves described in the exercise pdf.

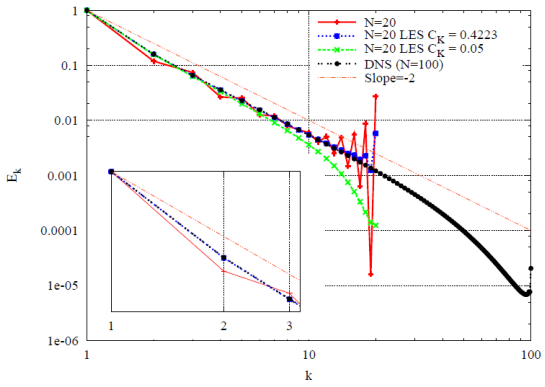


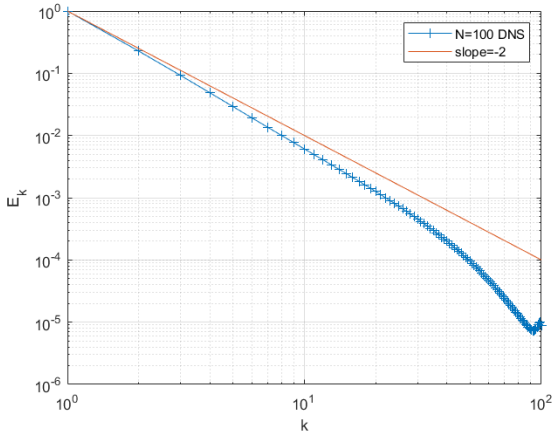
Figure 3: Energy spectrum of the steady-state solution of the Burgers equation for $N = 20$ (with spectral eddy-viscosity model taking $m = 2$ and $C_K = 0.4523$) and $N = 100$ (DNS).

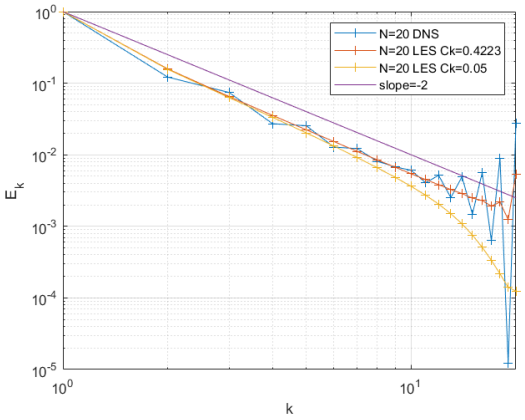
Then to see if my code is working we need to plot 5 curves:

- N=20
- The Direct Numerical Simulation with N=100
- The LES with the spectral eddy-viscosity model m=2, Ck = 0.4523 and N = 20
- The LES with the spectral eddy-viscosity model m=2, Ck = 0.05 and N = 20
- The plot of the slope of -2

5 RESULTS DISCUSSION

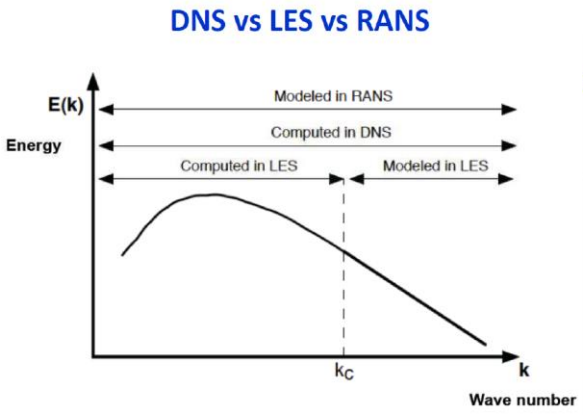
To be easier to see I have computed separately the case N=100 (on the left) and the others.

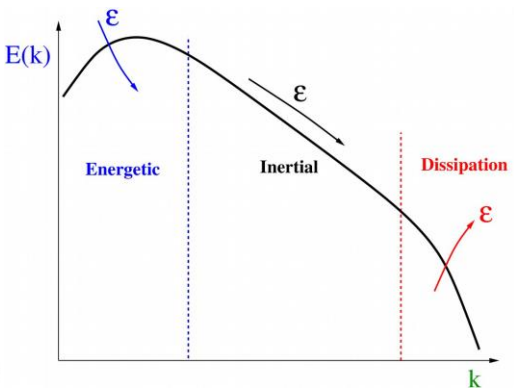




The direct numerical simulation is diverging at the end for both cases N=20 and N=100.

With all the models we can observe the diminution of the spectral energy. We can clearly see a part of the energy cascade. The decreasing energy is showing the passage from the large scale to the small scales.





To see what's happening we need to take a look at the theory . we clearly see for the LES case on N = 20 when we use a smaller Ck the dissipation effect is happening for lower k values . This is logical according to the Kolmogrov energy spectrum :

$$E(k) = C_K \varepsilon^{2/3} k^{-5/3}$$

We can also see the limitation of the LES computation near the dissipation range. The fact is that the DNS curve show an increase on the slope with the increasing k value witch is according with the spectral energy curve see in course.

6 CONCLUSION

The Burger equation was really a challenging code because of the complex numbers involved. The work on this subject allowed me to see more in practice the issues with the turbulence modeling. As the energy cascade as been shown in the results and it was also a good point to be able to see the roles of the convective and diffusive terms. The assignment as highlighted the difference between the LES and RNS model.

7 ANNEX

7.1 Code of Burger equation

```
%eX4 bURGER EQUATION
%aUTEUR bOULOGNE qUENTIN
clc;
clear ;
close all;

%% Input data

Re = 40;
N = 20;
C1 = 0.01;
Ck = 0.05;  %Kolmogorov constant
m=2;
LES = 0;  %0 desactivation of LES (Turbulence) 1 Activation
dt=(C1*Re)/(N^2);

Toler_conv=1e-6;

%% Without turbulence
LES = 0;
Ck = 0.05;
E1=CalcultationofBurgerequation(Ck,LES,N,Re,dt,m,Toler_conv,0.001);
```

```
%% LES turbulence Ck=0.4223
LES = 1;
Ck = 0.4223;
E2=CalcultationofBurgerequation(Ck,LES,N,Re,dt,m,Toler_conv,0.001);

%% LES turbulence Ck=0.05
LES = 1;
Ck = 0.05;
E3=CalcultationofBurgerequation(Ck,LES,N,Re,dt,m,Toler_conv,0.001);



%% N=100
LES = 0;
Ck = 0.05;
E4=CalcultationofBurgerequation(Ck,LES,100,Re,dt,m,1e-5,0.001);
N=100;
for k=1:N
    K(1,k)=k;
    Slp_1(k)=1/(k^2);
end
figure(1);

loglog(K,E4,'-+',K,Slp_1);
grid;
xlabel('k');
ylabel('E_k');
legend('N=100 DNS','slope=-2');

%% Post treatement
N=20;
for k=1:N
    K20(1,k)=k;
    Slp_2(k)=1/(k^2);
end
figure(2);

loglog(K20,E1,'-+',K20,E2,'-+',K20,E3,'-+',K20,Slp_2);

grid;
xlabel('k');
ylabel('E_k');

legend('N=20 DNS','N=20 LES Ck=0.4223','N=20 LES Ck=0.05','slope=-2');
```

7.2 Code calculation of Burger equation

```
function E = CalcultationofBurgerequation(Ck,LES,N,Re,dt,m,Toler_conv,C1)
Re=40;
dt=(C1*Re)/(N^2);
u0 = zeros(N,1);
K = zeros(N,1);
```

```
for k = 1:N
   K(k,1) = k;
   u0(k) = 1/abs(k);
end
t = 1;
u(:,t) = u0(:);
Diffusion=zeros(N,1);
Convection=zeros(N,1);
for k=1:N
    if LES==0 % Without LES
    Diffusion(k,1)=k^2*(1/Re)*u(k);
    end
    if LES==1% With LES
    Vt_infinite=0.31*(5-m)/(m+1)*sqrt(3-m)*Ck^(-3/2);
    Vt_star=1+34.5*exp(-3.03*N/k);
    Ekn=u(N)*conj(u(N));
    Vt=Vt_infinite*sqrt(Ekn/N)*Vt_star;
    Diffusion(k,1)=k^2*(1/Re+Vt)*u(k);
   end

    for p=-N+k:N
        q=k-p;
        if q~=0 && p~=0
            if q<0
                uq=conj(u(-q));
            else
                uq=u(q);
            end
            if p<0
                up=conj(u(-p));
            else
                up=u(p);
            end
            Convection(k)=Convection(k)+up*q*1i*uq;
        end
    end
end
Sum_ini(:,1)=Diffusion(K)+Convection(K);

t = t+1;
u(:,t) = u(:,t-1);

%% Solution

converging_condition = 0;
while converging_condition == 0
Diffusion=zeros(N,1);
Convection=zeros(N,1);
for k=1:N
    if LES==0
    Diffusion(k,1)=k^2*(1/Re)*u(k,t);
    end
    if LES==1
```

```
   Vt_star=1+34.5*exp(-3.03*N/k);
   Vt_infinite=0.31*(5-m)/(m+1)*sqrt(3-m)*Ck^(-3/2);
   Ekn=u(N,t)*conj(u(N,t));
   Vt=Vt_infinite*sqrt(Ekn/N)*Vt_star;
   Diffusion(k,1)=k^2*(1/Re+Vt)*u(k,t);
  end

   for p=-N+k:N
      q=k-p;
      if q~=0 && p~=0
         if q<0
            uq=conj(u(-q,t));
         else
            uq=u(q,t);
         end
         if p<0
            up=conj(u(-p,t));
         else
            up=u(p,t);
         end
         Convection(k)=Convection(k)+up*q*1i*uq;% C calculation
      end
   end
end
    Sum(K,1)=Diffusion(K)+Convection(K);
   u(K,t+1) = u(K,t)-dt*Sum(K,1);
   u(1,t+1) = 1;
   if max(abs(u(:,t+1)-u(:,t)))<Toler_conv
      E(K) = u(K,t+1).*conj(u(K,t+1));% enegie calculation only if the error is low in off
      converging_condition = 1; % Exiting condition
   else
      t = t+1;
      Sum_ini(K) = Sum(K);
   end
end
end
```