PDF Eulerian Method: An Exploration Based on The Works of Thomas D. Dreeben & Stephen B. Pope

Shashwat Patnaik (0289 1591)

Abstract

The modeling of Probability Density Function (PDF) evolution equations within the context of turbulent flows is explored. The primary objective is to develop and verify a solver using base cases such as Homogeneous Isotropic Turbulence, Pure Shear, and Axisymmetric Expansion, with a focus on extending the model to address complex flow scenarios involving wall effects. The report comprehensively explores turbulence modeling, emphasizing Reynolds-Averaged Navier-Stokes (RANS) equations and the limitations of traditional closures based on the Boussinesq approximation. Drawing inspiration from the works of Pope, additional physics is introduced into the models, particularly employing a generalized Langevin model combined with a model for viscous transport. The PDF model is presented, characterizing turbulent fluctuations as a vector-valued stochastic process, and specific case models are detailed, including the discretization and set-up. The iterative solver, based on the Runge-Kutta-4 scheme, is described, and the verification process is outlined using a pure diffusion PDE case. The discussion highlights the convergence with expected values in the Homogeneous Isotropic Turbulence case and acknowledging discrepancies with DNS data. The work contributes valuable insights into the challenges and advancements in modeling turbulent flows using the Eulerian PDF method, supported by references to seminal works by Dreeben, Pope, and Campos.

1 Problem & Objective

The problem is how to model the PDF evolution equations using existing closures and implementing a solver that has been verified using 3 base cases such as HIT, Pure Shear, and Axisymmetric Expansion. The problem is how to apply the model to more complicated flow cases such as those involving wall effects and more involved changes. A consequence of modeling more complex flow cases is that a solver that exhibits exceptional computational complexity is desirable.

2 Introduction

2.1 Turbulence Modeling

The traditional Reynolds decomposition of the fluid velocity and pressure can be represented as follows:

$$U_i = \langle U_i \rangle + u_i$$

$$P = \langle P \rangle + p$$

Here, $\langle U_i \rangle$ is mean velocity and $\langle P \rangle$ is mean pressure which can be expectations based on PDF, or spatial, temporal, or ensemble averages. The fluctuating velocity u_i and pressure p can be defined as the difference between the instantaneous and averaged quantities. Thus, the evolution of the mean velocity can be computed through the Reynolds-Averaged Navier-Stokes equation:

$$\frac{\partial \langle U_i \rangle}{\partial x_i} = 0$$

$$\frac{\partial \langle U_i \rangle}{\partial t} + \langle U_j \rangle \frac{\partial \langle U_i \rangle}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \langle P \rangle}{\partial x_i} + \nu \frac{\partial^2 \langle U_i \rangle}{\partial x_j \partial x_j} - \frac{\partial R_{ij}}{\partial x_j}$$

R_{ij} is the Reynold Stress Tensor which is equal to $\langle u_i u_j \rangle$. Now, the system only has four equations and ten unknowns, as RANS and PDF simulations do not resolve the fluctuating velocities. Hence, RANS and PDF models are made to provide an expression for R_{ij} using mean variables only, such that there is an equal number of equations to unknowns.

2.2 Need for PDF Models

The most popular RANS closures are those based on the Boussinesq approximation, like Eddy Viscosity Models, Differential Reynolds Stress Models, and Algebraic Reynolds-Stress Models. However, there is limited accuracy for flows where there is a sudden change or large magnitude of deformation rates, secondary motions, and three-dimensional effects when using models based on the Boussinesq approximation [2]. If more complex RANS models are used, then they exhibit more convergence for complex geometries [2]. Moreover, all of the RANS models failed to accurately model boundary layer separation. One way to improve the turbulence model is to introduce additional physics in the models. The paper by Pope achieves this by using a generalized Langevin model which is combined with a model for viscous transport [1]. This provides the exact treatment of viscous inhomogeneous effects [1]. It also introduces elliptic relaxation to compute G_{ij}, but for the cases presented in this report, it has frozen and there was no need to compute it [1]. However, it plays a critical role if the flow is inhomogeneous.

3 PDF Model

PDF methods characterize the turbulent fluctuation as a vector-valued stochastic process, using a one-point one-time PDF of fluctuating velocity. The paper by Pope provides an exact evolution equation for $\hat{f}$:

$$\frac{\partial \hat{f}}{\partial t} + V_i \frac{\partial \hat{f}}{\partial x_i} = \frac{1}{\rho}\frac{\partial \langle \vec{\varphi} \rangle}{\partial x_i}\frac{\partial \hat{f}}{\partial V_i} + \frac{\partial}{\partial V_i}[\hat{f}\langle \frac{1}{\rho}\frac{\partial p}{\partial x_i} - v\frac{\partial^2 U_i}{\partial x_j \partial x_j} | U(x,t) = V \rangle]$$

It is evident that the above equation can not be solved as the right-hand side of the equation contains unknown terms in the form of conditional expectations. Thus, a model whose domain is a field of probability densities of velocity, not of velocity itself is created. A stochastic characteristic frequency ω with a sample space variable Ω, is introduced for time-scale information. Where the Reynolds stress tensor is given by:

$$\langle U_i \rangle = \int V_i f \; dV \; d\Omega,$$

$$\langle u_i u_j \rangle = \int (V_i - \langle U_i \rangle)(V_j - \langle U_j \rangle) f \; dV \; d\Omega$$

The stochastic model for turbulent frequency is given by:

$$\langle \omega \rangle = \int \Omega f \; dV \; d\Omega$$

For inhomogeneous turbulent flows, Pope derives the following equation:

$$\begin{aligned}\frac{\partial \hat{f}}{\partial t} + V_i \frac{\partial \hat{f}}{\partial x_i} = v\frac{\partial^2 \hat{f}}{\partial x_i \partial x_i} + \frac{1}{\rho}\frac{\partial \langle \hat{\varphi} \rangle}{\partial x_i}\frac{\partial \hat{f}}{\partial V_i} - \frac{\partial^2}{\partial V_i \partial V_j} \\ \times [\hat{f}\langle v\frac{\partial U_i}{\partial x_k}\frac{\partial U_j}{\partial x_k} | U(x,t) = V \rangle] + \frac{\partial}{\partial V_i}[\hat{f}\langle \frac{1}{\rho}\frac{\partial p}{\partial x_i} | U(x,t) = V \rangle]\end{aligned}$$

Thus, the Probability density function using the Eulerian model becomes (based on Dreeben & Pope [1]):

$$\begin{aligned}\frac{\partial f}{\partial t} + V_i \frac{\partial f}{\partial x_i} = v\frac{\partial^2 f}{\partial x_i \partial x_i} + \frac{\partial f}{\partial V_i}\frac{1}{\rho}\frac{\partial \langle \vec{\varphi} \rangle}{\partial x_i} - \frac{\partial}{\partial V_i}[G_{ij}(V_j - \langle U_j \rangle) f] \\ + 2v\frac{\partial \langle U_j \rangle}{\partial x_i}\frac{\partial^2 f}{\partial x_i \partial V_j} + v\frac{\partial \langle U_i \rangle}{\partial x_k}\frac{\partial \langle U_j \rangle}{\partial x_k}\frac{\partial^2 f}{\partial V_i \partial V_j} + \frac{1}{2}C_0 \epsilon \frac{\partial^2 f}{\partial V_i \partial V_i} \\ + C_3 \langle \omega \rangle \frac{\partial}{\partial \Omega}[(\Omega - \langle \omega \rangle) f] + S_\omega \langle \omega \rangle \frac{\partial}{\partial \Omega}(\Omega f) + C_3 C_4 \langle \omega \rangle^2 \frac{\partial^2 (\Omega f)}{\partial \Omega \partial \Omega}\end{aligned}$$

Where the constant terms are given by [1]:

$$C_1 = 1.85 \quad C_2 = 0.63; \quad C_v = 1.4; \quad \gamma_S = 0.1; \quad C_T = 6.0; \quad C_L = 0.134; \quad C_u = 72.0$$

$$S_{ol} = C_{o2} - C_{ol}\frac{P}{\epsilon} + C_5 \max(0, 1 - \frac{P}{\epsilon})^3$$

$$C_{\omega 1} = 0.44; \quad C_{w2} = 0.73; \quad C_3 = 5.0; \quad C_4 = 0.25; \quad C_5 = 0.3$$

4 Case Specific Models

4.1 Homogeneous Isotropic Turbulence (HIT)

For HIT, the following terms are active, resulting in the modified PDE for this case:

$$\frac{\partial f}{\partial t} = \frac{1}{2}C_0\epsilon\frac{\partial^2 f}{\partial V_i \partial V_i} + C_3\langle\omega\rangle\frac{\partial}{\partial\Omega}[(\Omega - \langle\omega\rangle)f] + S_w\langle\omega\rangle\frac{\partial}{\partial\Omega}(\Omega f) + C_3C_4\langle\omega\rangle^2\frac{\partial^2(\Omega f)}{\partial\Omega\partial\Omega}$$

Certain terms, such as $C_0, \epsilon, C_3, \langle\omega\rangle, S_w, C_4$ are constant, which means that in the time marching scheme, these terms are unaffected by the partials. The other terms, however, must be differentiated properly using the chain rule. This is the derivation for the second term on the RHS, and the other terms follow this pattern similarly:

$$\begin{aligned} C_3\langle\omega\rangle\frac{\partial}{\partial\Omega}[(\Omega - \langle\omega\rangle)f] &= C_3\langle\omega\rangle\frac{\partial}{\partial\Omega}[\Omega f - \langle\omega\rangle f] \\ &= C_3\langle\omega\rangle\left(f + \frac{\partial f}{\partial\Omega}(\Omega - \langle\omega\rangle)\right) \quad \text{(assuming } \langle\omega\rangle \text{ constant w.r.t } \Omega) \end{aligned}$$

The implementation of this term can be seen as term2 in the code listing (see Listing X for function set-up). The other three terms are derived in a similar way, applying the chain rule, and hence are omitted from the report to save space.

```
def func():
    term1 = 0.5*C0*eps*(ddx(f,i,j,k,l,0,U) + ddx(f,i,j,k,l,1,V) + ddx(f,i,j,k,l,2,W))
    term2 = 1*f[i,j,k,l] + (O[l]-ome)*dx(f,i,j,k,l,3,O)
    term3 = Sw*ome*(f[i,j,k,l] + O[l]*dx(f,i,j,k,l,3,O))
    term4 = C3*C4*ome**2*(0 + dx(f,i,j,k,l,3,O) + (1+O[l])*ddx(f,i,j,k,l,3,O))
    term5 = -Gij[0,1]*(V[j]-Ui[1])*dx(f,i,j,k,l,0,U)
    tot_term = term1 + term2 + term3 + term4 + term5
    return tot_term
```

Listing X: Function set-up.

4.2 Pure-Shear & Axisymmetric Expansion

The case of shear or axisymmetric expansions means that a strain rate tensor S_{ij} (used interchangeably with the tensor G_{ij}) is imposed. Now the term:

$$-\frac{\partial}{\partial V_i}[G_{ij}(V_j - \langle U_j\rangle)f],$$

Exists and for Pure-Shear is differentiated as follows:

$$-\frac{\partial}{\partial U}[(V - \langle V\rangle)f],$$

which has been simplified to be the first partial derivative of the PDF in the direction of the U velocity space, multiplied by the fluctuation in the V velocity. As expected this will affect the component of the Reynolds stresses of which this shear is acting. Similarly, for axisymmetric expansion, the values of the strain-rate tensor is substituted into the equation prior to the differentiation as the case is known to be simplified.

5 Discretization/Set-up

The quantity that is being modeled is a probability density function (PDF) which consists of 5 variables: that is the velocity space components U, V, W, as well as Ω and a time component. There are 4 so-called "spatial" dimensions which means that the PDF is 4-dimensional. The differential evolution equation proposed by Dreeben & Pope [1] acts on each and every component of this matrix.

5.1 Initialization

There exists two choices for initialization of the PDF's and that is:

1. **Using a Gaussian distribution**
   This approach is an ideal scenario of which it is unlikely DNS simulation data will follow, however for validation purposes, such as applying the model to homogeneous isotropic turbulence (HIT), this is preferred. The initialization is as follows (Listing 1). It is also worth noting that the distribution is normalized to sum together to unity. The distribution is created using equal mesh spacing as well as having a mean centered around the origin with a standard deviation of 1. This means that if one chooses to visualize an arbitrary slice such as $f(:, \emptyset, \emptyset, :)$ it will result in a Gaussian which may be seen in Figure 1.

2. **Using DNS data to build the distributions**
   Another alternative is to use DNS data, which, when graphed, produces distributions similar to Figure 1 below, however, with a slight skew as it is based on actual experimental DNS data and it is known that HIT is not Gaussian in nature. The expectation is that the same results will be obtained, however slightly different magnitudes of the Reynolds stresses hovering around the expected value. As with the HIT verification case the expectation is that the quantity of $\langle u_i u_i \rangle / 2k$ will hover around 1/3.

```
f = np.zeros((n_grid,n_grid,n_grid,n_grid))
for i in range(n_grid): # U
    for j in range(n_grid): # V
        for k in range(n_grid): # W
            for l in range(n_grid): # O
                f[i,j,k,l] = gaussian1[i] * gaussian2[j] * gaussian3[k] * gaussian4[l]
f /= np.sum(f)
```

Listing 1: Initialization of 4D probability density function using 1D distributions for quantities U, V, W, Ω.

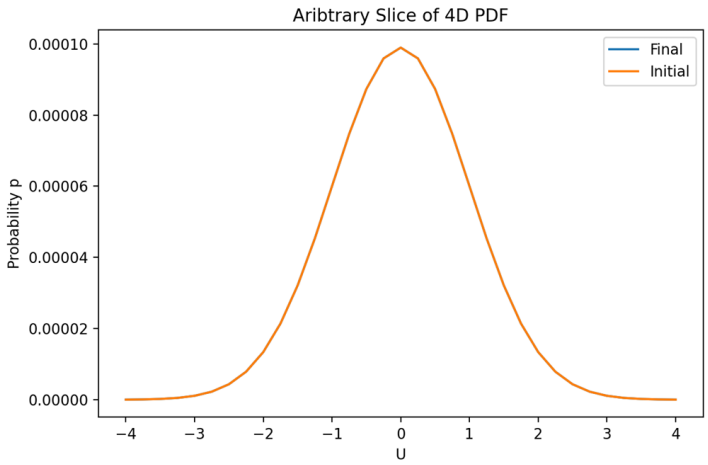

Figure 1: Slice of PDF using Gaussian distribution with the velocity space of the U component.

5.2 Differentiation

Due to the coarseness of the grid, it poses an interesting problem of how to compute finite differences at the end-points. First, the finite differences are computed to the second order using central differences using stencils:

$$\left.\frac{\partial u}{\partial x}\right|_{x_i} = \frac{u_{i+1} - u_{i-1}}{2\Delta x} - \frac{\Delta x^2}{6}\left.\frac{\partial^3 u}{\partial x^3}\right|_{x_i} + \dots$$

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{x_i} = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} - \frac{\Delta x^2}{12}\left.\frac{\partial^4 u}{\partial x^4}\right|_{x_i} + \dots$$

and at the end points using additional second-order finite differences. The initial approach was to set both the first and second derivatives to zero at the end points, which is what one should expect in an ideal case. However, the PDF evolution equation must be updated for each and every component of the 4D PDF, and therefore, the computational complexity increases vastly. Therefore, one-sided differences were also implemented in order to more accurately model the curvature and change of curvature in the distribution (see Listing 4 for structure). The truth is, that if one requires a different stencil in order to have well-behaved partials, then it should indicate that the domain, for example, velocity space, is not wide enough to capture all of the information.

5.3 Integration

Integration using the discrete distribution is required, and for the integration of a variable such as omega:

$$\langle \omega \rangle = \int \Omega f \; dV \; d\Omega,$$

It is done with the implementation concept shown in Listing 2 below.

```
for i in range(n_grid):
    ome += O[i]*np.sum(f[:,:,:,i])*dU*dV*dW*dO
```

Listing 2: Integration Snippet.

5.4 Deciding on a variable space

Upon inspection of the DNS data, the histogram bins of the distribution suggests that the majority of the information is contained within the bounds of $[-2.5, 2.5]$, and by virtue of this fact, the domain is initialized to $[-4, 4]$ in order to add a safety factor. Hence, the domain is defined by:

$$U \in [-4, 4]$$

$$V \in [-4, 4]$$

$$W \in [-4, 4]$$

$$\Omega \in [-4, 4]$$

And these bounds are used for both integration and differentiation. A good balance between performance and accuracy was found to be using an equally-spaced grid of 21 points. Ideally the distribution should be continuous and be constructed using even more points, however the code exhibits performance proportional to a complexity $\mathcal{O}(N^4)$.

5.5 Vectorizing the solver code

In an attempt to speed up the code, instead of using nested for loops, the derivatives may be vectorized in hopes of being able to use additional grid points and achieve increased accuracy. Prior to optimization the code structure is in line with Listing 3 below. The code in Listing 3 may now be modified to exclude the 4 loops, which are replaced by a single loop. This loop will compute the derivative of the function f with respect to a desired quantity. Listing 4 shows the modification to the second-derivative function which enables the computation of all derivatives with respect to the velocity space in a single computation. Using this form, the coefficients used in the Runge-Kutta-4 scheme may be computed all at once, and hence the 4 nested for-loops in Listing 3 may be removed.

5.6 Mean Field Estimation

The PDF model requires mean quantities like $\langle U_j \rangle$ and $\langle \omega \rangle$ to evaluate terms like the drift in the G_{ij} term and frequency evolution terms. In this Eulerian implementation, these moments were calculated directly by integrating over the computed PDF f using numerical quadrature (as conceptually shown for $\langle \omega \rangle$ in Listing 2). It is noted that for related Lagrangian particle methods, such as those described by Dreeben & Pope [1] which were studied in the context of extending this work, obtaining smooth mean fields from

```
for i in range(n_grid): # U
    for j in range(n_grid): # V
        for k in range(n_grid): # W
            for l in range(n_grid): # O
                k1 = func()
                f[i,j,k,l] += k1*dt/2

                k2 = func()
                f[i,j,k,l] = f[i,j,k,l] - k1*dt/2 + k2*dt/2

                k3 = func()
                f[i,j,k,l] = f[i,j,k,l] - k2*dt/2 + k3*dt

                k4 = func()
                f[i,j,k,l] -= k3*dt

                f[i,j,k,l] += (k1+2*k2+2*k3+k4)*dt/6
```

Listing 3: Nested for-loop structure prior to code optimization.

```
def ddx(f,i,j,k,l,dim,V):
    dx = V[1]-V[0]
    if i-1 < 0:
        return (2*f[i,j,k,l]-5*f[i+1,j,k,l]+4*f[i+2,j,k,l]-f[i+3,j,k,l])/dx**2
    if i+1 == len(V):
        return (2*f[i,j,k,l]-5*f[i-1,j,k,l]+4*f[i-2,j,k,l]-f[i-3,j,k,l])/dx**2
    ...
    if dim == 0: return (f[i+1,j,k,l]-2*f[i,j,k,l]+f[i-1,j,k,l])/dx**2
    ...
```

Listing 4: Modification to second-derivative function which enables the computation of all derivatives with respect to the velocity space in a single computation. The triple dot contains a repetition of code with respect to a different dimension.

scattered particle data requires regression techniques. Dreeben & Pope [1] (§A.2) detail a two-stage non-parametric method using kernel estimates followed by local least-squares fitting to compute mean fields and their derivatives at grid nodes from particle data. An adaptation involving quadratic least squares fitting was developed as part of the broader project context for potential use with Lagrangian methods.

5.7 Boundary Conditions

Appropriate boundary conditions are essential for both the phase-space domain and the physical domain (especially for inhomogeneous flows).

Phase-Space Boundaries (Implemented for Homogeneous Cases): The results presented in this report for homogeneous flows utilized boundary conditions at the limits of the computational domain in the U, V, W, Ω phase space (e.g., at ± 4). Computing derivatives at these boundaries used specific stencils. Tests were performed comparing simple zero-derivative conditions with second-order one-sided differences (results summarized in Table 1 and Table 2 from the verification study). Forcing the PDF to zero far from the center (Dirichlet condition) was achieved by ensuring the domain was sufficiently wide and checking overall probability conservation.

Physical Wall Boundary Conditions (Attempted for Channel Flow Extension): Extending the model to wall-bounded inhomogeneous flows like channel flow requires fundamentally different physical boundary conditions at the wall ($y = 0$). The approach detailed by Dreeben & Pope [1] provides a physically-based model for particle-wall interaction consistent with the no-slip condition. This involves (see Figure 2):

- Tracking particle paths near the wall using a stochastic model (Brownian motion + mean velocity).
- Probabilistically determining if a particle's path has intersected the wall within a time step.

- For interacting particles: Resetting the velocity to zero ($\mathcal{U}_i = 0$).
- For interacting particles: Resetting the turbulent frequency ω by sampling from a specific distribution (Gamma distribution) whose parameters depend on near-wall gradients (e.g., $\langle\omega\rangle \propto [(2k)^{1/2}]'$).

Implementing these complex, stochastic boundary conditions within the Eulerian framework, or ensuring the Eulerian solution near the wall is consistent with these conditions, proved to be a significant challenge. While attempted as part of the effort to simulate channel flow, instabilities and divergence issues prevented obtaining stable results with these boundary conditions coupled with the inhomogeneity models (Section 8). The simpler phase-space boundary treatments used for the homogeneous results are insufficient for accurately capturing wall effects.

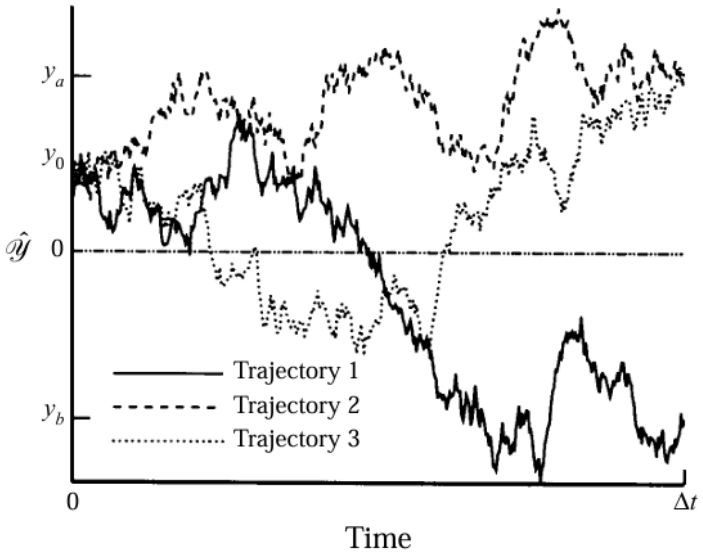

Figure 2: Illustration of possible particle trajectories near a wall, requiring distinct boundary treatment depending on whether the wall is encountered (Trajectories 1 and 3 involve wall contact). From Dreeben & Pope (1998) [1], Figure 1.

6 Iterative Solver

The solver consists of an outer loop that marches forward in time using a Runge-Kutta-4 scheme. This is visible in Listing 3 above. The function 'func()' is on the form shown in Listing 5. A time step Δt is decided through the diffusion limit and should be less than or equal to the grid-spacing squared:

$$\Delta t \leq \Delta x^2$$

The issue, however, is that due to the computational complexity of using a multi-dimensional distribution, the grid spacing is quite coarse. And although a time step is chosen to be in accordance with theory, it appears as if the time step is too large, causing the conservation of substance to break. Hence the time step is chosen to be an order of magnitude smaller than what the theory suggests.

6.1 Pure Diffusion Verification Case

To verify the correct implementation of the finite differences and methodology, a pure diffusion PDE will be evolved in time to verify the code. A simple diffusion equation PDE is defined by:

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

In order to confirm the implementation, one expects a certain scalar quantity to diffuse over time. Hence, an artificial 1D PDF is implemented containing a square bump and the expectation is that this initial state will even out over time. If this is the case, without losing information, then one can deduce that the finite differences are implemented correctly. See Figure 3 for the initial and final states.

```
def func():
    term1 = 0.5*C0*eps*(ddx(f,i,j,k,l,0,U) + ddx(f,i,j,k,l,1,V) + ddx(f,i,j,k,l,2,W))
    dx_0 = dx(f,i,j,k,l,3,0)
    term2 = C3 * ome * (f[i,j,k,l] + dx_0*(0[l]-ome))
    term3 = Sw * ome * (f[i,j,k,l] + dx_0*0[l])
    term4 = C3*C4*ome**2*2*dx_0 + 0[l]*ddx(f,i,j,k,l,3,0)
    term5 = -(Gij[0,1]*V[j]-Gij[0,1]*Ui[1])*(dx(f,i,j,k,l,0,U))
    tot_term = term1 + term2 + term3 + term4 + term5
    return tot_term
```

Listing 5: Equation 3.4 from Dreeben & Pope [1] in a functional form creating the basis for the PDF evolution equation.

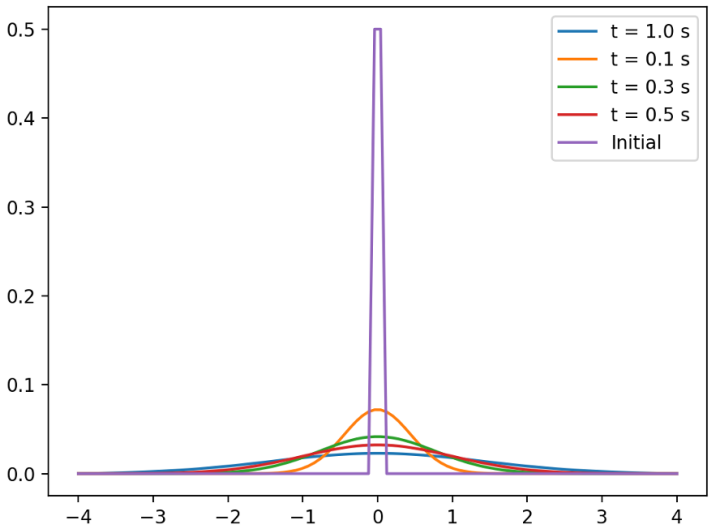

Figure 3: A square wave pulse at an initial time with an arbitrary scalar on the vertical axis and domain on the horizontal axis. Four lines reveal the evolution of the simple diffusion equation from zero to one second. A time step of 1e-4 is used.

Before time marching, the distribution is normalized, which suggests that at the end of the iterations, it should have conserved its state and integration should yield a value of one. See Table 1 below for different grid sizes and time steps.

Table 1: Changing Boundary Conditions and the Effect on Accuracy

Grid Size	Time Step	Total Time	End-Derivs	Conservation	Dirichlet
100	1e-4s	1 s	One-sided	0.988	No
100	1e-4s	1 s	One-sided	0.991	Yes
100	1e-4s	1 s	Zero	0.991	No
100	1e-4s	1 s	Zero	0.991	Yes

Furthermore, looking at Table 2, the effect of changing the grid size is varied and again the main output of interest is ensuring the conservation of substance. If the conservation is ill-behaved, it might suggest that when the iterative solver is marched forward in time toward 10^1 seconds and beyond, it will no longer be representative of the underlying mathematical models.

The time step was varied and the conclusion can be made that the proportionality between the time step and grid size from above holds more firm at a much finer grid, perhaps 500 points or above, which is unreachable with the computational resources available at this time. Hence the Dirichlet boundary conditions are enforced with a velocity and omega space that is large enough to enclose the bulk of the information stored in the distribution and a time step that is an order of magnitude smaller than what

Table 2: Changing Grid Size and the effect on accuracy

Grid Size	Time Step	Total Time	End-Derivs	Conservation	Dirichlet
5	1e-4s	1 s	Zero	0.955	Yes
11	1e-4s	1 s	Zero	0.984	Yes
21	1e-4s	1s	Zero	0.989	Yes
100	1e-4s	1 s	Zero	0.991	Yes

the theory might suggest.

7 Results

This section details the results obtained from applying the implemented Eulerian PDF solver to three homogeneous test cases: Homogeneous Isotropic Turbulence (HIT), Pure Shear Flow, and Axisymmetric Expansion. ** Note: In our model, $G_{ij} = S_{ij}$. For all simulations, the grid level for the domain is set to be 31, resulting in a PDF size of 31^4.

7.1 Homogeneous Isotropic Turbulence (HIT)

Description: HIT refers to the situation where the turbulent flow is statistically the same everywhere within the fluid domain. Thus, the probability density function in the Eulerian model is the same in the respective position. Hence, all the terms involving differentiation with respect to "x" become 0. Moreover as there is no shear acting, $S_{ij} = G_{ij}$ can be written as:

$$S_{ij} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Thus the Eulerian PDF model equation simplifies to:

$$\frac{\partial f}{\partial t} = \frac{1}{2} C_0 \epsilon \frac{\partial^2 f}{\partial V_i \partial V_i} + c_3 \langle \omega \rangle \frac{\partial}{\partial \Omega} [(\Omega - \langle \omega \rangle) f] + S_\omega \langle \omega \rangle \frac{\partial}{\partial \Omega} (\Omega f) + C_3 C_4 \langle \omega \rangle^2 \frac{\partial^2 (\Omega f)}{\partial \Omega \partial \Omega}$$

Figures:

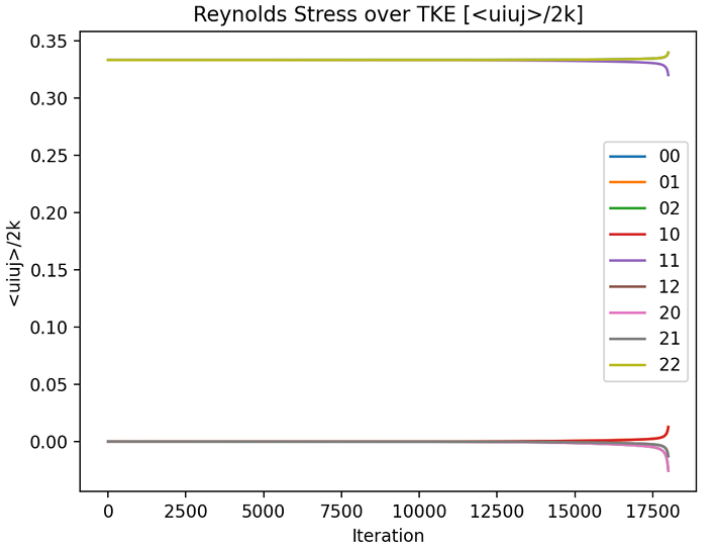

Figure 4: Reynolds Stresses for HIT Case.

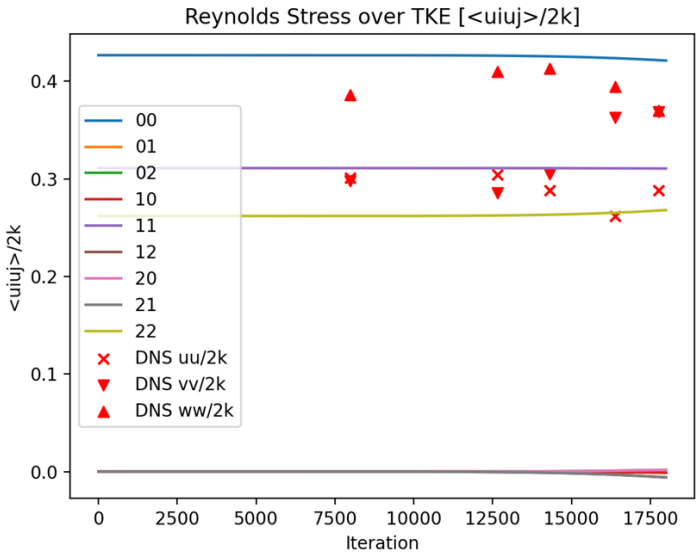

Figure 5: HIT evolution until solver encounters numerical errors plotted as solid lines and DNS data from John Hopkins Database [3].

Discussion: Using RK4 iterating to 18 seconds, with time step 1e-3, we get the plot for the Reynolds stress tensor shown in Figure 4. At a certain point the behavior of the Reynolds stress components suddenly change which occurs by virtue of the turbulent kinetic energy and Reynolds stress components approaching zero; the erratic behavior of the plot is due to numerical errors in the division as both approach their limit. For the HIT case, the theoretical Reynold stress tensor is:

$$\frac{R_{ij}}{2k} = \begin{pmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix}$$

It is evident from Figure 4 that all the diagonal components suggest convergence toward the $\frac{1}{3}$ value, and all the off-diagonal terms are 0.

Additionally, DNS data can be pulled into the graph to verify the Reynolds stress components at different time steps. Look at Table 3 for data from the John Hopkins Turbulent Database [3] with 8192^3 grid points.

Table 3: DNS Results for Forced Homogeneous Isotropic Turbulence 8192^3 Grid points

Snapshot	1	2	3	4	5
Time	3.7952	2.3938	4.2912	4.9124	5.3300
u	2.2943	2.2695	2.1744	1.9751	2.1729
v	2.1533	2.2458	2.2983	2.7367	2.7766
W	3.0881	2.9087	3.1131	2.9739	2.7889
k	3.7679	3.7120	3.7931	3.8429	3.8693
$uu/(2k)$	0.30446	0.30117	0.28855	0.26210	0.28836
$vv/(2k)$	0.28574	0.29802	0.30498	0.36316	0.36845
$ww/(2k)$	0.40980	0.38599	0.41311	0.39464	0.37009

As it turns out, the DNS data is not representative of the exact theory which contains a diagonal matrix of $\frac{1}{3}$. Instead it deviates quite significantly (e.g., 128^3 data yields $[0.3879, 0.2907, 0.3154]$ for diagonal components). Hence it is fair to assume that if the simulation values are close to one-third, the implementation is correct. Figure 5 shows that while the simulation was initialized from a Gaussian distribution (unlike the non-Gaussian nature of real turbulence), there are similarities in the trends towards isotropy compared to DNS, although quantitative differences exist and the components evolve differently.

7.2 Pure Shear Flow

Description: From HIT, we can verify model validity for homogeneous isotropic flows. Now, we impose a constant shear on the flow to test the validity of the model in non-isotropic flows. Now G_{ij} is not 0, thus the following model equation is used (homogeneous terms of the full PDF equation):

$$\frac{\partial f}{\partial t} = -\frac{\partial}{\partial V_i}[G_{ij}(V_j - \langle U_j \rangle)f] + \frac{1}{2}C_0\epsilon\frac{\partial^2 f}{\partial V_i \partial V_i} + c_3\langle\omega\rangle\frac{\partial}{\partial\Omega}[(\Omega - \langle\omega\rangle)f] + S_\omega\langle\omega\rangle\frac{\partial}{\partial\Omega}(\Omega f) + C_3C_4\langle\omega\rangle^2\frac{\partial^2(\Omega f)}{\partial\Omega\partial\Omega}$$

The strain-rate tensor is set to be:

$$S_{ij} = G_{ij} = \begin{pmatrix} 0 & S & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Figures:

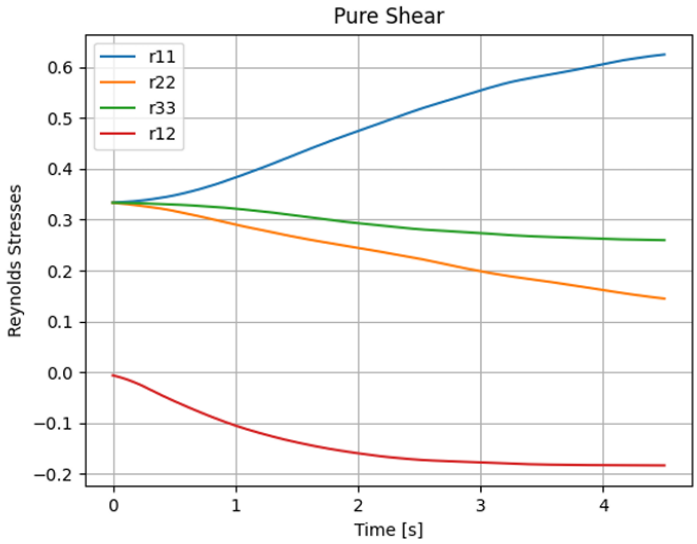

Figure 6: Reynolds Stresses for Pure Shear Case with $G_{ij} = 1$ ($S = 1$).

Discussion: Using RK4 iterating toward 4+ seconds, with time step 1e-4, and S set to 1, the plot for the Reynolds stress tensor obtained is shown in Figure 6. Comparing Figure 6 and Figure 7 (from Campos [2]), it is clear that there is a resemblance between the results which is reassuring. However, the solver exhibited forms of instability already after a time of 4.5 s, hence, the end result is not clear. It appears as if the first diagonal component of the Reynolds stress (r_{11}) will exceed that of 0.6 which is present in Campos's results. The expectation is that it might temporarily move above and hopefully stabilize at the expected value. A remedy for the behavior of the divergence is that a finer grid spacing and time step is used; however, even with the vectorized code it takes an unreasonable amount of time.

7.3 Axisymmetric Expansion

Description: Axisymmetric Expansion is similar to Pure Shear Flow, except G_{12} is not active now, and only diagonals terms are active in G_{ij}. The strain-rate tensor is set to be:

$$S_{ij} = G_{ij} = \begin{pmatrix} -S & 0 & 0 \\ 0 & \frac{1}{2}S & 0 \\ 0 & 0 & \frac{1}{2}S \end{pmatrix}$$

The same PDF Eulerian Model equation as used in Pure Shear is applied.

Figures:

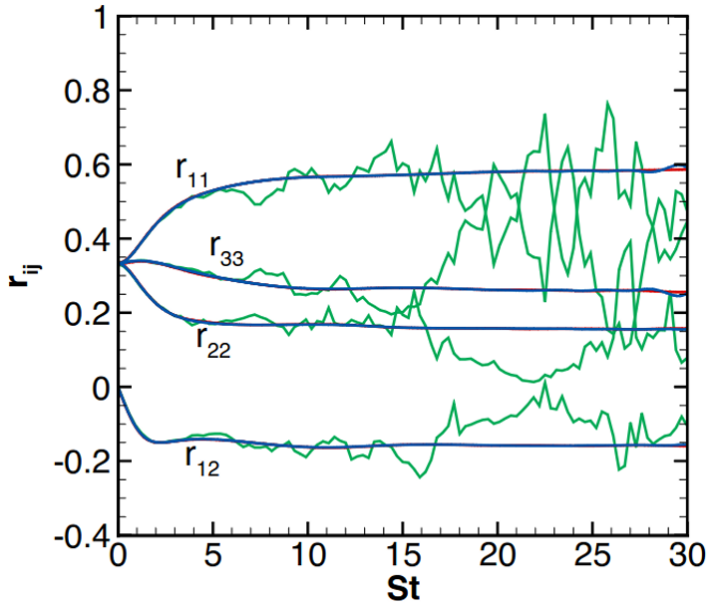

Figure 7: Reynolds Stresses for Pure Shear Case from Campos [2].

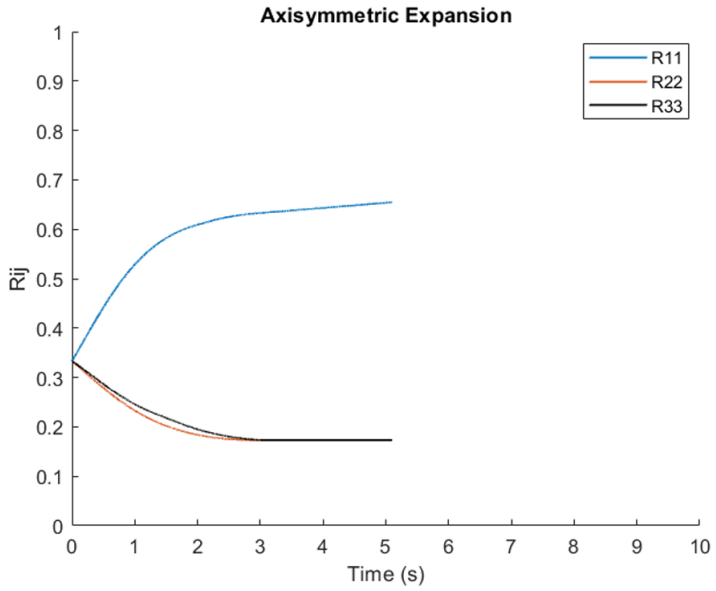

Figure 8: Reynolds Stresses for Axisymmetric Expansion with $S = 1$.

Discussion: Using RK4 iterating toward 5 seconds, with time step 1e-4, and S set to 1, the plot for the Reynolds stress tensor obtained is shown in Figure 8. After the solver exceeded 5 seconds the simulation blew up as k became extremely large, and then all values became NaN. Comparing with results from Campos [2] in Figure 9, there is a similarity in the trends before the instability. For Axisymmetric Expansion the time step was very critical. Any time step above 1e-4 leads the simulation to blow up and unable to converge. Figure 10 shows the simulation with time step $1e-2$; soon after 1 sec, the Reynolds stress begins to increase or decrease drastically and at a certain point, all the values are NaN.

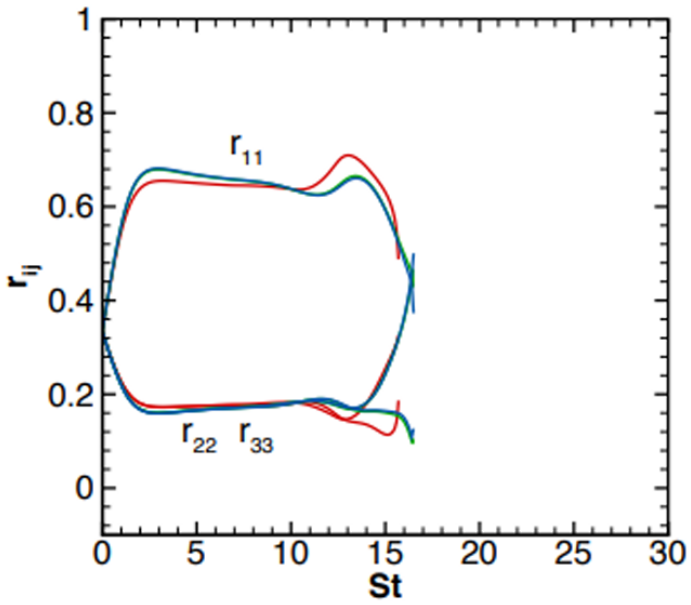

Figure 9: Reynolds Stresses for Axisymmetric Expansion Case from Campos [2].

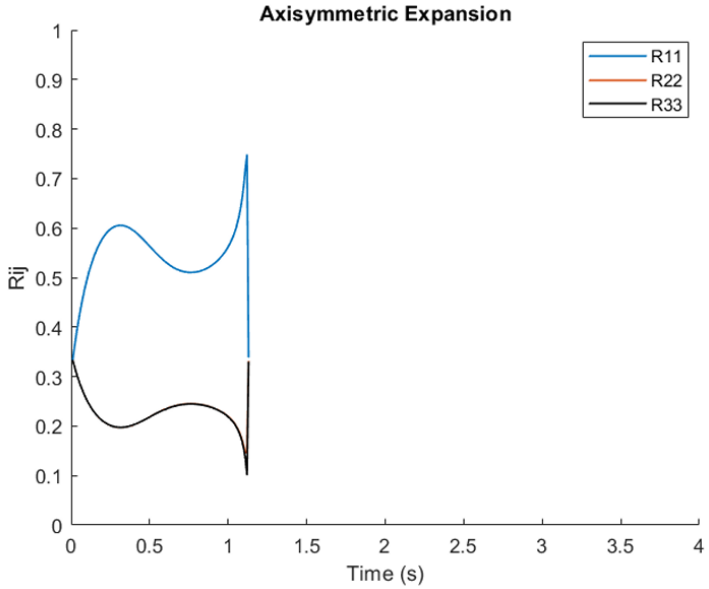

Figure 10: Reynolds Stresses for Axisymmetric Expansion with $S = 1$. Large dt failed.

8 Challenges and Limitations

The development and verification of the Eulerian PDF solver for homogeneous flows, and particularly the attempts to extend it towards inhomogeneous channel flow, highlighted significant challenges:

- **Computational Cost:** The $\mathcal{O}(N^4)$ scaling of the Eulerian approach in the 4D phase space severely limited grid resolution ($N \leq 31$) even for homogeneous cases. Extending to inhomogeneous channel flow multiplies this cost across a physical grid, making direct Eulerian discretization potentially infeasible without massive parallel resources.
- **Grid Resolution (Phase & Physical Space):** Coarseness in phase space impacts accuracy. Achieving sufficient resolution simultaneously in phase space and near-wall physical space for channel flow is extremely demanding.
- **Numerical Stability & Stiffness:** The explicit RK4 scheme necessitated very small time steps ($\Delta t \ll \Delta x^2$) for stability, especially in strained flows which became unstable. Divergence was encountered when incorporating inhomogeneity effects. This suggests stiffness related to model terms (G_{ij}, frequency evolution) or wall interactions.
- **Inhomogeneity Modeling (G_{ij}):**** Implementing and stabilizing the elliptic relaxation model for G_{ij}, required to capture near-wall anisotropy and inhomogeneity effects as described by Dreeben & Pope [1], proved difficult and contributed significantly to solver divergence issues when moving beyond the simplified homogeneous setup (where G_{ij} was fixed or set to S_{ij}).
- **Wall Boundary Conditions**:** Implementing physically correct boundary conditions for the PDF at solid walls is a major challenge for inhomogeneous flows. While the complex stochastic particle reflection/resetting mechanism detailed by Dreeben & Pope [1] (Figure 2) was attempted, successfully integrating and stabilizing these conditions within the Eulerian framework alongside the inhomogeneity models proved problematic and was a key factor preventing stable channel flow simulations. The phase-space BCs used for homogeneous results are insufficient.
- **Verification & Validation:** Rigorous verification is difficult due to the lack of analytical solutions for the full PDF equation. Validation against available DNS/reference data for homogeneous cases showed qualitative agreement but also discrepancies.

9 Future Improvements and Alternative Eulerian Methods

Based on the experience gained and the goal of simulating inhomogeneous wall-bounded flows like channel flow, future work should prioritize the following:

- **Stabilize Inhomogeneous Solver Components**:**
  - Focus effort on achieving a stable and robust implementation of the spatial derivative terms and their coupling with phase-space terms.
  - Develop a stable solver for the elliptic relaxation equation governing G_{ij}, essential for near-wall anisotropy [1].
  - Successfully implement and stabilize the complex Dreeben & Pope wall boundary conditions [1] (stochastic particle analogy: reflection, velocity/frequency reset) or identify/develop alternative robust wall treatments compatible with the Eulerian framework. This is critical.
- **Improve Numerical Robustness & Accuracy:**
  - Adopt implicit or IMEX time-stepping schemes specifically designed to handle the stiffness arising from model terms (mixing, frequency evolution, G_{ij}) and potential wall interactions.
  - Implement higher-order, low-dissipation finite difference/volume schemes for both phase-space and physical-space derivatives to improve accuracy, especially near walls.
  - Ensure rigorous conservation and positivity preservation.
  - Conduct comprehensive verification using methods like MMS for the full inhomogeneous solver, including boundary conditions.
- **Enhance Computational Performance:**

  - Develop efficient parallel algorithms (MPI-based domain decomposition) capable of handling the combined physical and phase-space grids required for inhomogeneous flows.

- **Explore Alternatives**:**
  - Investigate alternative Eulerian formulations (e.g., Method of Moments, consistent EMCF) that might offer better scaling or stability characteristics for high-dimensional inhomogeneous problems.
  - Evaluate alternative closure models (mixing, pressure-strain, frequency) specifically validated for near-wall turbulent flows.

References

References

[1] DREEBEN, T. D., & POPE, S. B. (1998). Probability density function/Monte Carlo simulation of near-wall turbulent flows. *Journal of Fluid Mechanics*, 357, 141–166. https://doi.org/10.1017/s0022112097008008

[2] Campos, Alejandro. (2016). Advances in Structure-Based Modeling of Turbulent Flows. Stanford Digital Repository.

[3] Johns Hopkins Turbulence Databases (JHTDB). (n.d.). Turbulence.pha.jhu.edu. Retrieved December 22, 2023, from https://turbulence.pha.jhu.edu/Isotropic8192.aspx