

Guided Research Report

Turbulence Modelling with JAX-SPH

MohammadAmin SaeidyPour



TUM Uhrenturm

Guided Research Report

Turbulence Modelling with JAX-SPH

MohammadAmin SaeidyPour

Abstract

Differentiability of (Computational Fluid Dynamics) solvers can increase the capabilities and usability of them, eg. gradient-based optimisation or integration of machine-learning blocks. There are some instances of differentiable CFD solvers but, none are based on Lagrangian formulations. In this research, a differentiable CFD solver is developed that solves Taylor-Green Vortex problem based on a Lagrangian formulation. The formulation is the transport-velocity Smoothed Particle Hydrodynamics (SPH) formulation proposed by [Adami et al.(2013)]. As a by-product of that development, this research also introduces some techniques to improve the accuracy of the results with regard to those in [Adami et al.(2013)]. One is to set the reference density (ρ_0) as the minimum of density field in each time step. ρ_0 is a numerical coefficient to control the convergence in SPH simulations. The technique is named "adaptive reference density" because it is novel and it emphasises the change of ρ_0 as opposed to keeping it constant. The other is to divide ρ_0 by a constant named "pressure-ratio" to balance the magnitudes of momentum, reference and background pressures. Momentum pressure is the general, known pressure. Reference pressure is a term that is calculated with ρ_0 and the speed of sound. Background pressure is also a term introduced by [Adami et al.(2013)]. Because the exact calculation of energy spectrum in turbulence simulations are important, the solver was further developed to calculate energy spectrum exactly for velocities distributed on a regular or irregular grid. Fast Fourier Transformation (FFT) is the formulation that calculates the energy spectrum exactly for velocities of a regular, uniformly placed points. For irregular ones, the current most accurate formulation is introduced by [Shi et al.(2013)]. That is second-order Moving Least Squares (MLS) formulation. The developed solver is validated for both FFT and MLS formulations.

Contents

Abstract	v
1 Introduction	1
2 Method	3
2.1 State-of-the-art concerning the development of the JAX-SPH code based on the transport-velocity formulation of [Adami et al.(2013)]	3
2.1.1 Governing equations	3
2.1.2 Numerical time-integration	5
2.1.3 Points regarding implementation in JAX	6
2.2 State-of-the-art concerning the exact calculation of energy spectrum using FFT and second-order MLS methods proposed by [Shi et al.(2013)]	7
2.2.1 State-of-the-art concerning FFT	8
2.2.2 State-of-the-art concerning MLS	8
3 Results and discussion	11
3.1 Development of a code in JAX with the transport-velocity formulation in [Adami et al.(2013)]	11
3.1.1 Validation with figure 4 of [Adami et al.(2013)]	11
3.1.2 Improvement of transport-velocity formulation: Balancing the effect of momentum and background pressure by introducing adaptive-reference-density technique and pressure-ratio constant	14
3.1.3 Improvement of results with improving pressure's accuracy	15
3.2 Development of a code in JAX for exact calculation of the energy spectrum for velocities on regular and irregular grids	16
3.2.1 Calculation of energy spectrum with FFT and validation with the results from [Shi et al.(2013)]	18
3.2.2 Calculation of energy spectrum with MLS	22
4 Conclusions	27
4.1 Conclusions of part 1, development of a JAX-SPH code and validation of its results with the results from [Adami et al.(2013)]	27
4.2 Conclusions of part 2, development of a JAX code that calculates energy spectrum exactly for velocities from regular or irregular grids	28
Bibliography	29

1 Introduction

Turbulence modelling with JAX-SPH is the title of this guided research. Turbulence is one of the main regimes that categorises the behaviour of a flow. It is mainly applicable when the behaviour of the flow is not laminar. Hence, a mixture between layers is expected.

Many researchers have focused on turbulence modelling for a long time because, in industry and nature, laminar flow is hardly seen. As flows are also present almost everywhere, the applications vary widely. Consequently, there is still very active interest in improving the speed and accuracy of the simulations.

JAX-SPH in the title highlights the programming language and the formulation that are used for the simulation. SPH is the acronym for "Smoothed Particle Hydrodynamics". This formulation is a Lagrangian formulation because it divides the flow field into particles and then, follows the behaviour of each particle.

JAX is a programming language with great features. One of its main advantages is that the developed solver in JAX is differentiable. In fact, the core novelty and aim for this research is to develop the first differentiable, Lagrangian CFD solver in JAX. In the literature, there exists CFD solvers in JAX. For example, [Kochkov et al.(2021)] have integrated a convolutional neural network into the famous projection method and they have developed it in JAX. As another example, [Bezgin et al.(2023)] have developed a CFD solver in JAX for compressible two-phase flows. The formulation in that research is based on finite-volume method and on a Cartesian grid. To the best of our knowledge, up to now, no differentiable CFD solver has been developed that is based on a Lagrangian formulation. Therefore, the main aim of this research is novel.

In order to develop the solver, a problem had to be chosen to be solved with that specific programming language and that particular formulation. Taylor-Green Vortex (TGV) problem is a famous decaying vortex problem that can span all three regimes of laminar, transition and turbulence. Moreover, considerable amount of study has been done on the problem with all three theoretical, numerical and experimental approaches. Therefore, the problem was perfect to benchmark a solver with.

Among all studies on TGV with SPH, this research specifically focuses on the recreation of the results from [Adami et al.(2013)]. The reason is that the previous research has been completed in the same chair as the one that this research is done in.

During this research, two techniques are developed that have improved the results in [Adami et al.(2013)]. [Adami et al.(2013)] reports that particles align along their stagnation lines at the beginning of the simulations and this can lead to additional error in the simulation. [Adami et al.(2013)] solves the previous problem by repeating the same simulation however, with different initial particle configuration. Their first initial particle configuration is uniform Cartesian. After the first simulation ends, [Adami et al.(2013)] repeats the same simulation with the particle configuration of the last time step of the first simulation. [Adami et al.(2013)] reports that because the particle configuration of the "pre-run" is irregular but also, compatible with the simulations, particles do not align along their stagnation lines anymore. Hence, the errors are reduced.

While the use of a "pre-run" can solve the problem, the solution is not appealing due to its computational cost. Therefore, this research introduces two techniques that, when combined, improve the results and eliminate the need of a pre-run. One technique is to set the reference density equal to the minimum density of the field in each step and the other is to divide it by a constant to control the impact of different pressures (more on this in chapter 2 and 3).

Those two techniques are not introduced before to the best of our knowledge and based on a review on the literature. Because [Adami et al.(2013)] first introduces the transport-velocity formulation to avoid particle clustering, the articles that have cited [Adami et al.(2013)] were particularly focused on in the review.

This research also aims to develop a code in JAX that exactly calculates energy spectrum. Energy spectrum is a diagram that shows the distribution of energy throughout different wavenumbers. This spectrum is particularly important for shock wave and turbulent flow analysis. In a turbulent flow, the behaviour of smallest vortexes are important due to their impact on the dissipation of energy. Hence, their exact modelling in a simulation can improve the accuracy of the whole simulation. The effect of those vortexes on dissipation can be studied with energy spectrum. Moreover, comparison of energy spectrum of different flows (or a flow with theoretical, numerical or experimental data) reveals the accuracy of the simulation specifically when a closure model is used to simulate turbulence.

[Shi et al.(2013)] have studied and presented different methods to calculate energy spectrum. According to [Shi et al.(2013)], Fast Fourier Transformation (FFT) can lead to exact calculation of energy spectrum for velocities on a regular, uniform grid. It also claims that its proposed second-order Moving Least Squares (MLS) can calculate energy spectrum with the least introduced error.

Like the literature on the use of reference density, the literature on the energy-spectrum calculation is extensive and diverse because it is calculated in different research fields (not even for a flow field and not even from velocities). However, some of the articles that have cited [Shi et al.(2013)] have used FFT and MLS for the calculation of energy spectrum and their results show good agreement with the expected energy-spectrum function. Therefore, it seems safe to conclude that FFT and second-order MLS are currently the most accurate methods to calculate energy spectrum.

In the following chapter, the method of this research is presented.

2 Method

Chapter method contains the state-of-the-art (governing equations and numerical method) and points concerning the implementation of the code in JAX. This chapter is divided into two sections, each representing the corresponding state-of-the-art of each part of the research. The first part is the state-of-the-art concerning the development of the JAX-SPH code based on the transport-velocity formulation of [Adami et al.(2013)]. The next part is the state-of-the-art concerning the exact calculation of energy spectrum using FFT and second-order MLS methods proposed by [Shi et al.(2013)].

2.1 State-of-the-art concerning the development of the JAX-SPH code based on the transport-velocity formulation of [Adami et al.(2013)]

2.1.1 Governing equations

In this subsection, governing equations are introduced. The governing equations are the famous Navier-Stokes (NS) equations. NS are based on the conservation principles. The main principles in physics are mass, momentum and energy conservation. In this work, the energy conservation is not considered (which leads to a distribution of temperature).

The Smoothed Particle Hydrodynamics (SPH) version of [Adami et al.(2013)] is weakly compressible SPH. Although the formulation is weakly compressible SPH, with the use of a technique, mass conservation is exactly and automatically held. Therefore, mass conservation is not considered in governing equations as well. The technique is to allocate a constant mass to each particle. With this technique and considering the fact that particles will not neither vanish nor be created (constant number of particles), the mass is conserved exactly.

While the mass is conserved exactly, the formulation of [Adami et al.(2013)] is weakly compressible SPH as stated above. There is no contradiction because it is the variation of density throughout the field that makes the formulation weakly compressible (and not incompressible, which is constant density throughout the field).

While variations of density can exist, the governing equations of weakly compressible SPH are based on the assumption that variations are less than one percent. In mathematics formulation, the previous assumption is:

$$\frac{\Delta\rho}{\rho} \leq 1\% \quad (2.1)$$

where ρ is the density. The momentum conservation in mathematical formulation, according to [Adami et al.(2013)], is:

$$\frac{\tilde{d}(\rho\mathbf{v})}{dt} = -\nabla p + \eta\nabla^2\mathbf{v} + \nabla\cdot(\rho\mathbf{v}(\tilde{\mathbf{v}} - \mathbf{v})). \quad (2.2)$$

In the above, the gravity acceleration is neglected because, in periodic TGV, the gravity acceleration is not considered. t is time, \mathbf{v} is the momentum velocity, p is the pressure, η is the dynamic viscosity and $\tilde{\mathbf{v}}$ is the transport velocity. The introduction of transport velocity is the core idea of [Adami et al.(2013)]. The tilde shows the transport variables. The material derivative of a moving particle with $\tilde{\mathbf{v}}$, $\frac{\tilde{d}}{dt}$, is defined as:

$$\frac{\tilde{d}(\bullet)}{dt} = \frac{\partial(\bullet)}{\partial t} + \tilde{\mathbf{v}}\cdot\nabla(\bullet). \quad (2.3)$$

The equation for momentum conservation in [Adami et al.(2013)] is obtained only by adding and subtracting the convective transport term to the usual conservation of equation of NS. The result is:

$$\frac{\partial(\rho v_i)}{\partial t} + \tilde{v}_j \frac{\partial(\rho v_i)}{\partial x_j} - \tilde{v}_j \frac{\partial(\rho v_i)}{\partial x_j} + v_j \frac{\partial(\rho v_i)}{\partial x_j} = [\dots]_i - \rho v_i \frac{\partial v_j}{\partial x_j}. \quad (2.4)$$

By rearranging the terms and combining them using the equation 2.3, equation 2.2 can be obtained. Whole transport-velocity formulation of [Adami et al.(2013)] solves equation 2.2.

In SPH formulations, when at each time step, the positions, the velocities and the accelerations of particles are known then, the whole system is defined and known. Positions are calculated from known velocities. Velocities are updated using the accelerations.

The equation of acceleration is:

$$\frac{d\tilde{v}_i}{dt} = \mathbf{a}_p + \mathbf{a}_\eta = \frac{1}{m_i} (\mathbf{f}_p + \mathbf{f}_\eta) = \frac{1}{m_i} \sum_j (V_i^2 + V_j^2) [-\tilde{p}_{ij} \frac{\partial W}{\partial r_{ij}} \mathbf{e}_{ij} + \frac{1}{2} (\mathbf{A}_i + \mathbf{A}_j) \cdot \frac{\partial W}{\partial r_{ij}} \mathbf{e}_{ij} + \tilde{\eta}_{ij} \frac{\mathbf{v}_{ij}}{r_{ij}} \frac{\partial W}{\partial r_{ij}}]. \quad (2.5)$$

The index i and j in equation 2.4 is different than the i and j in equation 2.5. In equation 2.4, i and j refer to coordinates (and summable for each term having the same index twice). However, i in equation 2.5 refers to the current particle and j is all particles that interact with the particle i *including* i itself. [Adami et al.(2013)] states that summation of j is done over all neighbouring particles of i and, hence, it may lead to misunderstanding that i itself must be excluded. But, it seems [Adami et al.(2013)] considers each particle a neighbour for itself because otherwise, the summations will be wrong.

In equation 2.5, V is the volume of each particle. The volume is calculated using the known mass (m_i) and the density of each particle ($V_i = \frac{m_i}{\rho_i}$). \tilde{p}_{ij} is calculated with:

$$\tilde{p}_{ij} = \frac{\rho_i p_j + \rho_j p_i}{\rho_i + \rho_j}. \quad (2.6)$$

p is pressure with \mathbf{a}_p and \mathbf{f}_p being the acceleration and force caused by pressure (and difference of momentum and transport velocities) respectively. In SPH, p is calculated with the equation of state and from the density. That equation is:

$$p = C^2(\rho - \rho_0) = p_0 \left(\frac{\rho}{\rho_0} - 1 \right) \quad (2.7)$$

where C is the speed of sound, which is a numerical coefficient to control the convergence in SPH. In this research, according to [Adami et al.(2013)], $C = 10U$, where U is the maximum initial velocity of Taylor-Green Vortex problem. ρ_0 and p_0 are reference density and reference pressure respectively. They both have the same function as the speed of sound's.

$\tilde{\eta}_{ij}$ is the averaged viscosity calculated by:

$$\tilde{\eta}_{ij} = \frac{2\eta_i \eta_j}{\eta_i + \eta_j}. \quad (2.8)$$

As η is the dynamic viscosity, \mathbf{a}_η and \mathbf{f}_η are the acceleration and force contributions of viscosity.

\mathbf{A}_i is $\rho_i \mathbf{v}_i (\tilde{\mathbf{v}}_i - \mathbf{v}_i)$. $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$ and $\frac{\partial W}{\partial r_{ij}} \mathbf{e}_{ij}$ is the gradient of W . W is the kernel function, which is responsible for bridging the discrete values in SPH to a continuum field. In [Adami et al.(2013)], the kernel function is quintic spline. According to [Xenakis et al.(2015)], the quintic spline for 2D is:

$$W(\mathbf{r}_i - \mathbf{r}_j, h) = \frac{1}{h^2} \frac{7}{478\pi} \begin{cases} (3-s)^5 - 6(2-s)^5 + 15(1-s)^5 & \text{if } 0 \leq s < 1 \\ (3-s)^5 - 6(2-s)^5 & \text{if } 1 \leq s < 2 \\ (3-s)^5 & \text{if } 2 \leq s < 3 \\ 0 & \text{if } 3 \leq s \end{cases} \quad (2.9)$$

\mathbf{r}_i is the position vector of particle i . $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ and $s = \frac{|\mathbf{r}_{ij}|}{h}$. h is the cutoff radius. Its value is chosen to be constant and equal to the initial spacing of particles.

The other important equation is for the update of positions. That is:

$$\frac{d\mathbf{r}_i}{dt} = \tilde{\mathbf{v}}_i. \quad (2.10)$$

The systems of equations are closed if the calculation of ρ and $\tilde{\mathbf{v}}$ are known. ρ is calculated as follows:

$$\rho_i = m_i \sum_j W_{ij}. \quad (2.11)$$

$\tilde{\mathbf{v}}_i$ is updated using the following equation:

$$\tilde{\mathbf{v}}_i(t + \delta t) = \mathbf{v}_i(t) + \delta t \left(\frac{d\mathbf{v}_i}{dt} + \mathbf{a}_{pb} \right). \quad (2.12)$$

\mathbf{a}_{pb} is:

$$\mathbf{a}_{pb} = \frac{\mathbf{f}_{pb}}{m_i} = -\frac{p_b}{m_i} \sum_j (V_i^2 + V_j^2) \frac{\partial W}{\partial \mathbf{r}_{ij}} \mathbf{e}_{ij}. \quad (2.13)$$

Because p_b is the added background pressure and mainly to complete the transport-velocity formulation, \mathbf{a}_{pb} and \mathbf{f}_{pb} are transport acceleration and force accordingly. In this research, reference pressure and background pressure are set equal based on the suggestion of [Adami et al.(2013)]. Therefore, both depend on the magnitude of ρ_0 .

δt is the time step size. It is calculated with the following equation:

$$\delta t = 0.25 * \text{minimum} \left(\frac{h}{C + |U|}, \frac{h^2}{\nu} \right) \quad (2.14)$$

where $\nu = \frac{\eta}{\rho}$.

2.1.2 Numerical time-integration

The numerical time-integration scheme of [Adami et al.(2013)] is as follows:

First, the following update is done to acquire the momentum and transport velocity at the next half step. The reason is that the calculation of forces can be done only once per each time step:

$$\mathbf{v}^{n+1/2} = \mathbf{v}^n + \frac{\delta t}{2m_i} (\mathbf{f}_p^{n-1/2} + \mathbf{f}_\eta^{n-1/2}) \quad (2.15)$$

$$\tilde{\mathbf{v}}^{n+1/2} = \mathbf{v}^{n+1/2} + \frac{\delta t}{2m_i} \mathbf{f}_{pb}^{n-1/2}. \quad (2.16)$$

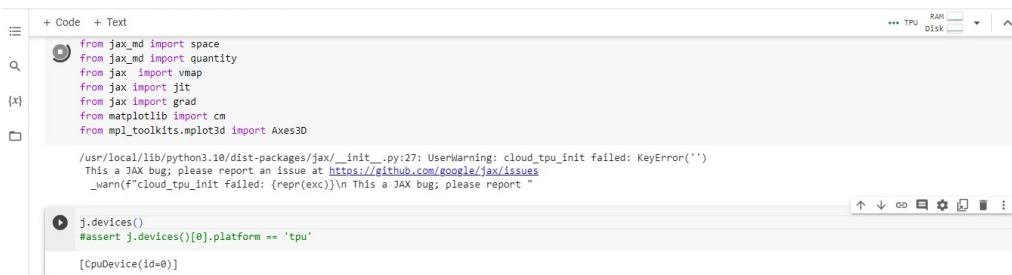
Then, the positions are updated:

$$\mathbf{r}^{n+1} = \mathbf{r}^n + \delta t \tilde{\mathbf{v}}^{n+1/2}. \quad (2.17)$$

At this point, the density and pressure will be updated with the new positions. Then, the forces at $n + 1/2$ is calculated using the new positions and the new velocities at $n + 1/2$. Subsequently, the velocities are calculated with:

$$\mathbf{v}^{n+1} = \mathbf{v}^{n+1/2} + \frac{\delta t}{2m_i} (\mathbf{f}_p^{n+1/2} + \mathbf{f}_\eta^{n+1/2}). \quad (2.18)$$

Therefore, now, \mathbf{r}^{n+1} , \mathbf{v}^{n+1} and accelerations at the current step's half ($n + 1/2$) are known and the calculations for the new time step can begin.



```
+ Code + Text
from jax_md import space
from jax_md import quantity
from jax import vmap
from jax import jit
from jax import grad
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D

/usr/local/lib/python3.10/dist-packages/jax/_init_.py:27: UserWarning: cloud_tpu_init failed: KeyError('').
This is a JAX bug; please report an issue at https://github.com/google/jax/issues
  _warn(f"cloud_tpu_init failed: {repr(exc)}\n This is a JAX bug; please report ")

j.devices()
#assert j.devices()[0].platform == "tpu"

[CpuDevice(id=0)]
```

Figure 2.1 Google Colaboratory labels the inability of JAX to use its TPUs as a bug in JAX.

2.1.3 Points regarding implementation in JAX

In order to get most out of the features of JAX, a solver should be written in vectorised format (operations on arrays) as much as possible. This was the part that needed creativity because no similar code in the literature could be found. JAX-MD library, which is the Molecular Dynamics library of JAX, has a short documentation with some examples. However, the examples in that documentation were not also very useful for this research.

It should also be noted that the format of the neighbor list is an important factor in the design of a vectorised solver. To create the neighbor list and also to implement the periodic boundaries, this research used the functions of JAX-MD. The format of the neighbor list in JAX-MD can be "Dense", "Sparse" or "Ordered sparse".

A dense neighbor list is a matrix with each particle as a row and each neighbor as a column. A sparse one is an array with two rows that has the index of particles on one row and the other row has the index of the corresponding neighbor particle. Therefore, in the first row, the index of each particle is repeated as many times as the number of its neighbors. An ordered sparse neighbor list, takes a particle into account only once either as a neighbor or as a main particle. This corresponds to creating a sparse list but, only for particles with the index that is smaller than the index of its neighbor.

By choosing each neighbor list format, a different design of a code is needed because each variable is broadcasted to the shape of the neighboring list and then, how they are combined will be different in each code.

In this research, first, a dense neighbor list was selected. The reason was that it was easier to code. Although easier, the code had considerable amount of redundant operations and therefore, it was slow.

Consequently, the sparse neighbor list was selected as the basis of the code. Ordered sparse one could also be selected which could lead to the half of the needed operations for sparse neighbor list. However, because some extra broadcasting was needed to write a fully vectorised code, it seemed sparse format could be the most optimal solution.

Because the code was developed fully based on array operations, there was no need to use extra supportive vectorising functions of JAX such as vmap function. Another feature that becomes available if a JAX code is developed in a vectorised way is the "Just-In-Time" compilation feature. This feature caches the compilation of a function. This improves the speed of the code further because when the input arrays are changed, JAX just replaces them in the previous cache of the function. Therefore, the output is calculated more quickly for each input.

Although JITting a JAX code can improve the speed, several caveats exist that prevent a JAX-SPH code to be JITed for some of its parts. One important caveat is that shapes of an array can change but, they should not be dependent on the values of other arrays. Because most intermediate arrays change shape according to the shape of the neighbor list, JITting them is not recommended (according to JAX's documentation, it may lead to indefinite behaviour even if correct results are observed during tests.).

It should also be mentioned that according to the Google Colaboratory, JAX cannot benefit from its TPUs. The errors shown by Google Colaboratory itself are in figures 2.1 and 2.2.

Although no error was returned when using the GPUs in the Google Colaboratory, no considerable speed-up was observed when switching from CPU to GPU. Because the whole code was written in scal-

```

+ Code + Text
--> 16 jax.tools.colab_tpu.setup_tpu()
D: /usr/local/lib/python3.10/dist-packages/jax/tools/colab_tpu.py in setup_tpu(tpu_driver_version)
37 def setup_tpu(tpu_driver_version=None):
38     """Returns an error. Do not use."""
--> 39     raise RuntimeError(textwrap.dedent(message))

RuntimeError:
As of JAX 0.4.0, JAX only supports TPU VMs, not the older Colab TPUs.

We recommend trying Kaggle Notebooks
(https://www.kaggle.com/code, click on "New Notebook" near the top) which offer
TPU VMs. You have to create an account, log in, and verify your account to get
accelerator support.
Once you do that, there's a new "TPU VM v3-8" accelerator option. This gives
you a TPU notebook environment similar to Colab, but using the newer TPU VM
architecture. This should be a less buggy, more performant, and overall better
experience than the older TPU node architecture.

It is also possible to use Colab together with a self-hosted Jupyter kernel
running on a Cloud TPU VM. See
https://research.google.com/colaboratory/local-runtimes.html
for details.
SEARCH STACK OVERFLOW

```

Figure 2.2 Google Colaboratory explicitly returns an error that JAX cannot support the use of its TPUs anymore.

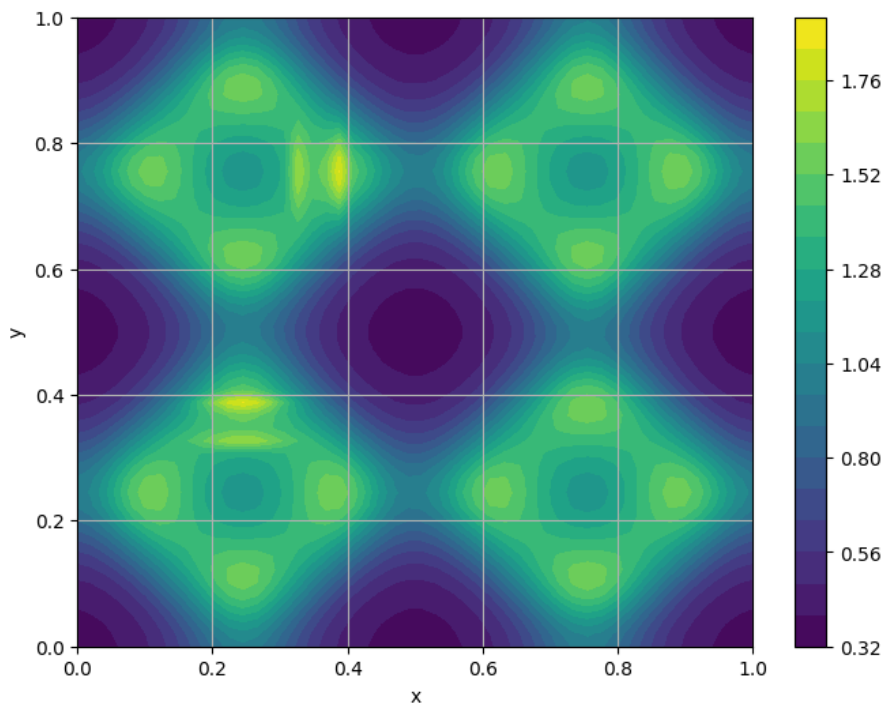


Figure 2.3 Density field is visibly miscalculated on 4 points due to a wrong calculation of distances between the particles. Those points are approximately $(x, y) = (0.25, 0.35), (0.25, 0.4), (0.35, 0.75)$ and $(0.4, 0.75)$.

able fashion, a considerable speed-up was expected upon switching from CPUs to GPUs (and further to TPUs). It seems this was also an issue of JAX with Google Colaboratory because JAX is responsible to automatically map the code to the dedicated device.

During this research, some other bugs from JAX and its libraries were also noticed. For example, an error in calculating distances between particles was seen that had lead to local discrepancies in all variables. One such discrepancy (inside the density field) is shown in figure 2.3. The bug could be avoided by simply changing the cutoff radius from $3.0h$ to $3.001h$.

2.2 State-of-the-art concerning the exact calculation of energy spectrum using FFT and second-order MLS methods proposed by [Shi et al.(2013)]

According to the literature review in chapter 1 and [Shi et al.(2013)], Fast Fourier Transformation (FFT) exactly calculates energy spectrum for velocities that are distributed on a uniformly placed, regular grid.

The formulation is discussed in subsection 2.2.1. For irregular grid, the current most accurate formulation to calculate energy spectrum is second-order Moving Least Squares (MLS). Its formulation is presented in subsection 2.2.2.

2.2.1 State-of-the-art concerning FFT

The standard method to calculate energy spectrum ($E(k)$, where k is the wavenumber) in regular grids is through the application of FFT to the velocity field. If $\mathbf{v}(\mathbf{r})$ is the velocity field of a d dimensional domain L^d , the FFT integral of $\mathbf{v}(\mathbf{r})$ is defined as:

$$\mathbf{V}(\mathbf{k}) = \frac{1}{L^d} \int e^{-i\mathbf{k}\cdot\mathbf{r}} \mathbf{v}(\mathbf{r}) d\mathbf{r} \quad (2.19)$$

with $d\mathbf{r} = dx dy dz$ in 3D and $d\mathbf{r} = dx dy$ in 2D. Using the previous, the velocity spectrum is:

$$E(\mathbf{k}) = \frac{1}{2} |\mathbf{V}(\mathbf{k}) \cdot \mathbf{V}^*(\mathbf{k})| \quad (2.20)$$

where $\mathbf{V}^*(\mathbf{k})$ is the complex conjugate of $\mathbf{V}(\mathbf{k})$ and $\mathbf{k} = (k_x, k_y, k_z)$ is the wavenumber vector. Finally, the energy spectrum in isotropic problems is calculated with:

$$E(k) = 4\pi k^2 \langle E(\mathbf{k}) \rangle \quad (2.21)$$

for 3D problems and

$$E(k) = 2\pi k \langle E(\mathbf{k}) \rangle \quad (2.22)$$

for 2D ones where $\langle \dots \rangle$ symbolises averaging over a thin spherical (3D) or circular (2D) shell of radius $k = |\mathbf{k}|$.

Fourier transformation (equation 2.19) is calculated using the command "jax.numpy.fft.fft" in JAX.

2.2.2 State-of-the-art concerning MLS

In irregular grids, MLS interpolates the velocities on a corresponding regular grid. Afterwards, FFT exactly calculates the energy spectrum for the acquired velocity field. Therefore, the only source of error is MLS.

According to [Shi et al.(2013)], MLS stems from the Taylor expansion of the velocity of a point on the irregular grid with regard to the points on the corresponding regular grid. The following equation is the mathematical formulation of the previous:

$$f_j = f_i + \sum_{n=1}^{+\infty} \frac{\partial^n f_i}{n!} (\mathbf{r}_j - \mathbf{r}_i)^n \quad (2.23)$$

where f is either the velocity in the x or y direction. Index j refers to a point on the irregular grid and index i denotes the effective points on the corresponding regular grid for the expansion. Therefore, there are known values (velocities on irregular grid) and unknown values (which are the interpolated velocities on the corresponding regular grid with their derivatives). This can form a system of equations. However, the Taylor expansion extends to infinity. Therefore, it is crucial to omit upon a certain limit. The upper limit for the sum will be selected as α , which will be the order of accuracy of MLS. Because this research focuses on the second-order MLS, here $\alpha = 2$.

By using a localised least-squares approach, the error E_i is calculated with:

$$E_i = \sum_{j=1}^N L_{ij} ((f_j)_\alpha - f_j)^2. \quad (2.24)$$

In the above, N is the number of points on the irregular grid. $(f_j)_\alpha$ is equation 2.23 that is summed only for the first α terms. L_{ij} is the localisation parameter that weights the contribution of each term to the total error. According to [Shi et al.(2013)], for SPH simulations, L_{ij} is calculated with the following:

$$L_{ij} = V_j w_{ij} \quad (2.25)$$

where w_{ij} is the kernel function of SPH and V is also the volume of each particle.

Based on the above, the goal is to minimise the error in each point or to make the approximation of the Taylor expansion as accurate as possible. Hence, the error's derivative with regard to f_i and its derivatives should become zero. This leads to a system of equations of size $SE \times SE$ for each point. SE is:

$$SE = \sum_{j=1}^{\alpha} \frac{(d-1+j)!}{(d-1)!j!} + 1 \quad (2.26)$$

with d being the dimension of the problem. The vectorised form of the system of equations for second-order MLS is as follows:

$$\begin{aligned} \sum_{j=1}^N V_j w_{ij} [f_{ij} + \sum_{n=1}^2 \frac{\partial^n f_i}{n!} \mathbf{r}_{ji}^n] &= 0 \\ \sum_{j=1}^N V_j w_{ij} [f_{ij} + \sum_{n=1}^2 \frac{\partial^n f_i}{n!} \mathbf{r}_{ji}^n] \mathbf{r}_{ji} &= 0 \\ \sum_{j=1}^N V_j w_{ij} [f_{ij} + \sum_{n=1}^2 \frac{\partial^n f_i}{n!} \mathbf{r}_{ji}^n] \mathbf{r}_{ji}^2 &= 0 \end{aligned} \quad (2.27)$$

where $f_{ij} = f_i - f_j$.

One important point in solving the above system of equations is that the condition of the coefficient matrix is dependent on the particle configuration. Therefore, it is possible that considerable errors for some points are observed. In order to reduce this error, this research uses the Tikhonov regularisation to stabilise the solution. Tikhonov regularisation is:

$$(A + \lambda I)f = b \quad (2.28)$$

where A is the coefficient matrix, λ is the regularisation constant, I is the identity matrix. f is the unknown matrix (here, f_i and its derivatives) and b the matrix of known values.

The results and the corresponding discussions of this research are presented in the next chapter.

3 Results and discussion

3.1 Development of a code in JAX with the transport-velocity formulation in [Adami et al.(2013)]

3.1.1 Validation with figure 4 of [Adami et al.(2013)]

In order to validate the developed code in JAX, comparisons will be made with figure 4 of [Adami et al.(2013)], specifically for $t = 0.2 s$. The reason is that the focus of this research is on Taylor-Green Vortex problem. In [Adami et al.(2013)], TGV is an example among several examples provided to show the robustness of the transport-velocity formulation. In the example concerning the TGV, [Adami et al.(2013)] has presented the results of a 2D TGV with the Reynolds number of $Re = 100$. [Adami et al.(2013)] has repeated the exact same case but, with another initial positions for the particles. Except for the previous two, no other results are provided for TGV. Because the developed code initialised particles into a uniform Cartesian grid, the results could only be compared with the results in figures 4 and 5 of [Adami et al.(2013)] (their first set of results).

Because figure 4 of [Adami et al.(2013)], specifically the snapshot for $t = 0.2 s$, has visual, distinctive characteristic, it was decided to start the validation with reproducing that figure (snapshot).

Because the result of SPH simulations, particularly the particle positions, depend on several, numerical (and not necessarily physical) quantities, several attempts were made to recreate the mentioned snapshot. Those attempts led to the discovery of two techniques that can be seen as improvements of the transport-velocity formulation proposed by [Adami et al.(2013)]. As discussed in chapter 1, the two improvements have not been proposed by any other article. Therefore, they can be considered as innovative improvements.

As mentioned above, [Adami et al.(2013)] has studied only one case for TGV but, the two sets of results have been presented. The only difference between the two sets is that the first set starts the simulation with particles on a cartesian grid with uniform spacing. However, the second set repeats the simulation exactly but, with placing the particles at the same positions as the end of the first set. The results of the first set of simulation are presented in figures 4 and 5 of [Adami et al.(2013)]. The results of second set however, are presented in figures 6, 7 and 8 of [Adami et al.(2013)].

The reason to repeat the simulation with the final particle positions of the previous run (pre-run) is to avoid particles from aligning along their stagnation lines at the beginning of the simulation and hence, to avoid noticeable error in the results. The two proposed improvements here also successfully prevent the particles from aligning along their stagnation lines and achieve the benefits of using the positions of a pre-run without actually using it.

In this subsection, first, the closest results to snapshot $t = 0.2 s$ of figure 4 of [Adami et al.(2013)] are presented and discussed. Afterwards, in subsection 3.1.2, the method to prevent stagnation lines and the corresponding results are presented and discussed. In the end, in subsection 3.1.3, the second improvement which can increase the accuracy of the results of subsection 3.1.2 is introduced.

Back to the main point of this subsection, the efforts started with the recreation of snapshot $t = 0.2 s$. The flow is initialised mainly with velocities because pressure in SPH simulations is supposed to be calculated with the equation of state. However, it should be noted that this research also used the analytical initial pressure of TGV for initialisation of pressure and although it is applied only for one step, difference in the acquired results was visible. The pressure which was used for initialisation was the following equation for $t = 0 s$ (based on [Taylor and Green(1937)]):

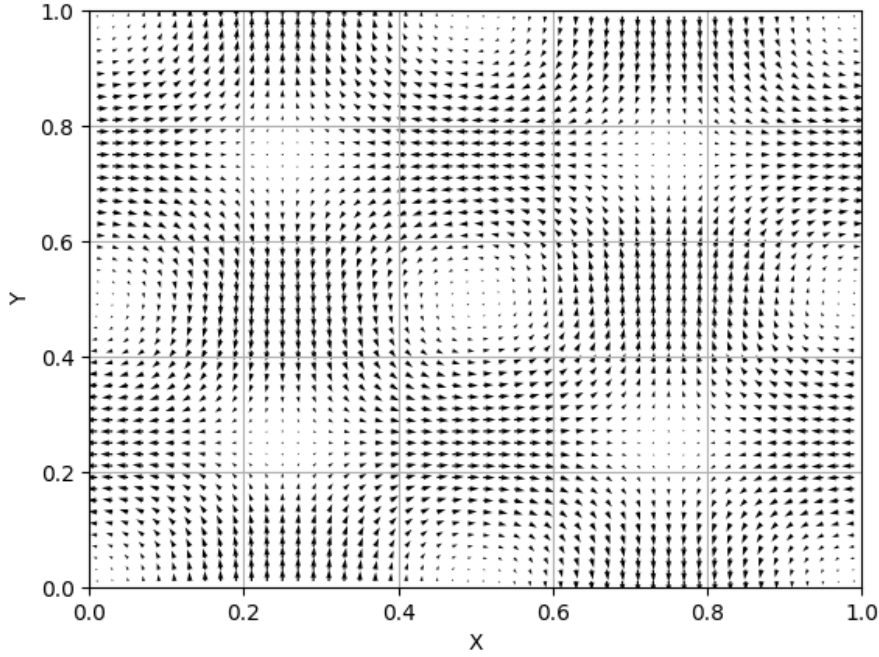


Figure 3.1 TGV's initial velocity field (vortexes)

$$P_{TGV} = \frac{\rho}{4.0} (e^{(bt)^2} (\cos(4\pi x) + \cos(4\pi y))). \quad (3.1)$$

$b = -8\pi^2 Re^{-1}$ is the decay rate of the vortexes. The initial velocity is the initial velocity of the analytical solution for the 2D TGV. The analytical solution for the velocity field is [Taylor and Green(1937)]:

$$u = -Ue^{bt} \cos(2\pi x) \sin(2\pi y) \quad (3.2)$$

$$v = Ue^{bt} \cos(2\pi y) \sin(2\pi x) \quad (3.3)$$

where U is the maximum, initial velocity of TGV. The visualisation of the velocity field is in figure 3.1.

The domain size in this case is a square with side length of 1. The domain is also bi-periodic (periodic in both perpendicular directions). The resolution is 50×50 , the same as the resolution selected for figure 4 of [Adami et al.(2013)].

One of the characteristics of the SPH simulations is that the result of the simulations, specifically the particle configurations, might depend on the numerical constants that do not result from physical features and equations. It was tried best to exactly recreate the figure 4 (naturally the snapshot for $t = 0.2 s$ because that is the meaningful snapshot of all four to be recreated). One of the closest results are in figure 3.2. As evident in figure 3.2, the particles are aligned more or less along stagnation lines. This figure is selected based on its resemblance to the desired snapshot. But, the problems caused by the alignment is not at its worst in this figure. One of the problems that this alignment creates is that it causes the density variation in the field to be above one percent (equation 2.1).

Equation 2.1 is an important assumption in weakly compressible SPH formulations. Hence, it should not be violated in order to avoid creation of more error in the simulations. Figure 3.3 shows one of the achieved configurations where very separated particle alignment has caused a severe violation of equation 2.1.

In the following subsection, one method to prevent particles from aligning along their stagnation lines and uphold the equation of 2.1 at least for TGV of $Re = 100$ is proposed. The method works even if the simulation is initialised with particles being on a uniform Cartesian grid.

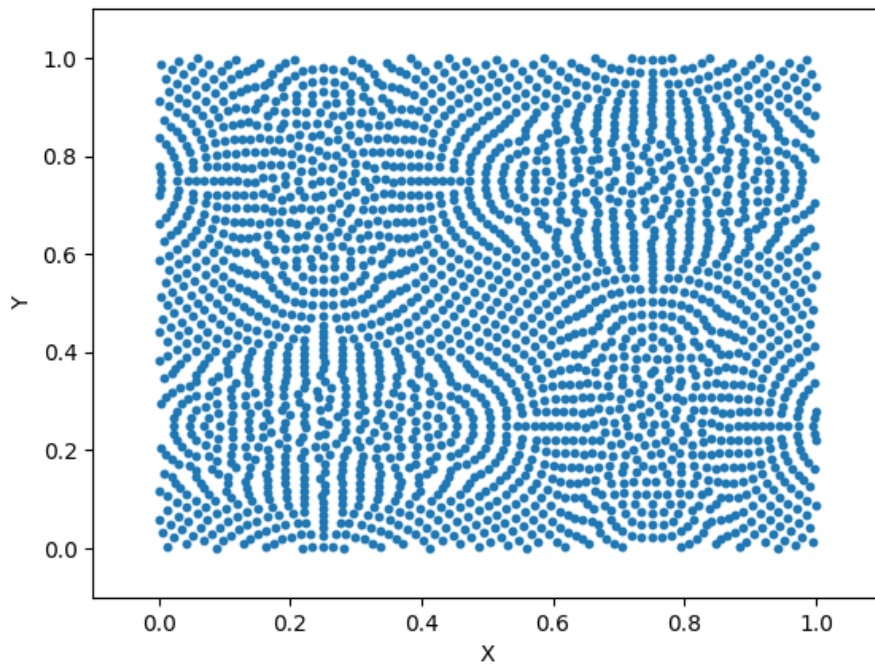


Figure 3.2 This snapshot is the closest recreated snapshot to figure 4 of [Adami et al.(2013)] for configuration at $t = 0.2 s$.

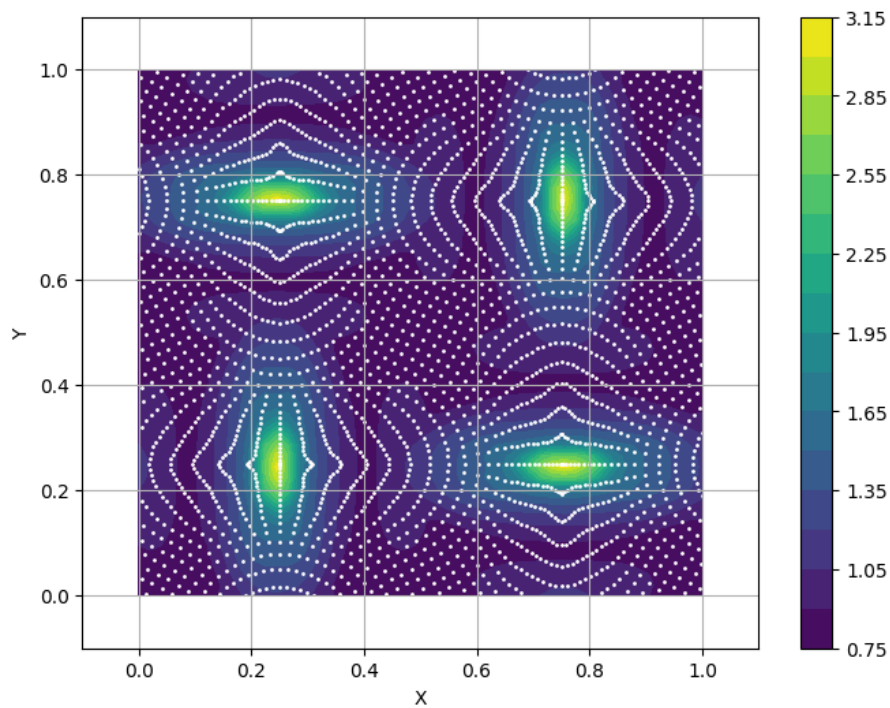


Figure 3.3 One of the configurations with the worst violation of basic assumption of weakly compressible SPH (equation 2.1) at $t = 0.2 s$. The contour is the density field while the white dots are particles.

3.1.2 Improvement of transport-velocity formulation: Balancing the effect of momentum and background pressure by introducing adaptive-reference-density technique and pressure-ratio constant

Based on the information in [Adami et al.(2013)], the main reason to introduce the formulation and particularly the background pressure, is to assure that the pressure field is positive throughout the domain. This can prevent clustering of particles in the standard SPH formulation. Therefore, in [Adami et al.(2013)], there are two important pressures. One pressure is the general pressure also known as momentum pressure and the other is the added background pressure.

Also in SPH formulations, reference density plays a role to stabilise the numerical convergence. Because in the equation of state, the reference density is deducted from the density, in any point of the domain that reference density has a value larger than the local density, the local pressure will be negative. In order to prevent the previous, it is proposed to set the reference density as the minimum of density of the field and adapt it throughout the simulation. Hence, at no time and no point, a negative pressure is obtained. This method can be called adaptive-reference-density method.

This effect can also be achieved by the proposed method of [Adami et al.(2013)], which is to increase the background pressure to an extent to assure that the above result is achieved. However, unlike the above method and as also mentioned by [Adami et al.(2013)], the size of time step must be adapted with the magnitude of background pressure because it affects the magnitude of its corresponding force, f_{pb} . f_{pb} is also used to integrate the movement of particles with. But, in the proposed time-step limits by [Adami et al.(2013)], no limit is proposed for p_b and, consequently, its magnitude does not directly have any effect on the size of the time step. Therefore, it seems adaptive-reference-density method is superior to simply increasing p_b .

Adaptive-reference-density method can be improved further if it is divided by a constant that is coined here as pressure-ratio constant. As the name suggests this constant is related to pressure ratios in a case. But, how?

If the variation of density is assumed to be zero (incompressible fluid instead of weakly compressible one), the calculated pressure from equation of state will also be zero. This causes the elimination of contribution of pressure to force in the acceleration equation (equation 2.5). As discussed also in [Adami et al.(2013)], ideally (if the particles would not cluster), the added contribution of difference between momentum and transport velocities (A in equation 2.5) is zero. Therefore, the only remaining contribution will be the one from the viscous force. This ideal case is accurate for TGV because in TGV, only viscous forces should be present. However, two points should be considered. The variation of density is not expected to be kept zero for all points and at all time steps. Therefore, the use of adaptive-reference-density method seems useful for all cases. The other point is that according to [Adami et al.(2013)], background and reference pressure are chosen to be equal (ie. $p_b = p_0 = C^2 \rho_0$). Hence, even in the ideal case, a large ρ_0 (reference density) leads to a larger magnitude of f_{pb} and, consequently, a larger displacement at each time step.

In order to reduce the magnitude of f_{pb} (displacement of particles) and to balance the magnitude of momentum pressure and background pressure, it is proposed to divide the adaptive reference density, which is selected to be the minimum value of density in the domain at each time step, by a constant. It is proposed to choose 2.0 for that constant. This halves the displacement of the particles (increasing the accuracy) and introduces a pressure with almost the same magnitude of background pressure to the domain, which is expected to further reduce the displacement of each particle at each time step and, hence, increase the accuracy. Because all forces are added up to the velocity of the previous step in the end, the momentum will still be conserved no matter of the size of the pressure-ratio constant. In fact, this corresponds to simply selecting a different value for reference density.

With these two changes, particles will not align along their stagnation lines anymore. Therefore, the density variation will remain below one percent. Moreover, the velocity field will maintain its form (similar to the initialisation velocity-field in figure 3.1). Therefore, with the technique proposed by this research, there will be no need for a pre-run to improve the results. The only important variable that does not improve is

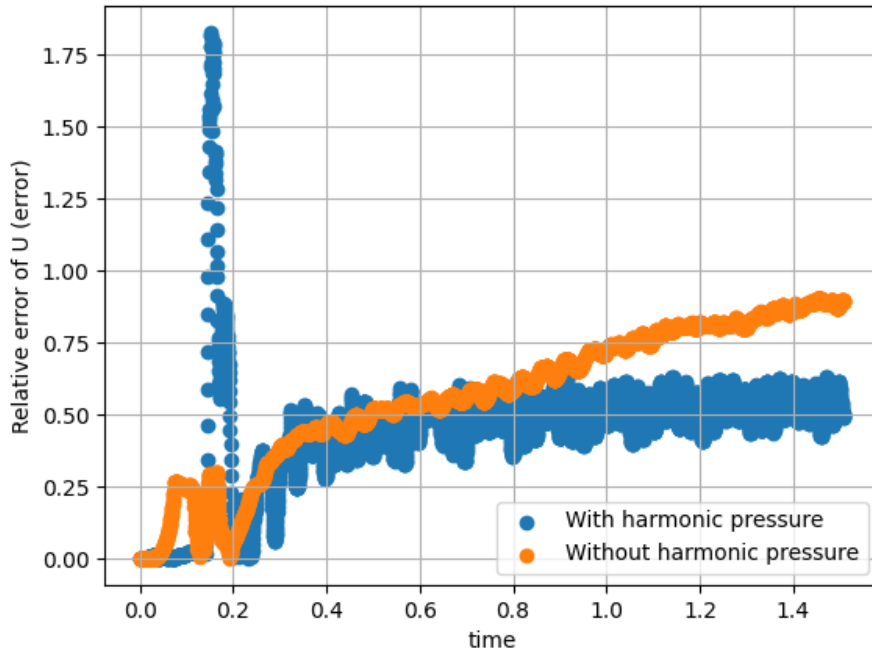


Figure 3.4 The relative error of maximum velocity, U , (equation 3.4) for the cases with and without the added harmonic pressure (equation 3.1) is shown.

the relative error of the decaying maximum velocity. The following equation is the mathematical definition of the relative error of the maximum velocity:

$$error = \left| \frac{\max(|\mathbf{v}_i(t)|) - Ue^{bt}}{Ue^{bt}} \right|. \quad (3.4)$$

In order to improve the relative error, it was decided to improve the pressure's accuracy. The resulted pressure from the SPH's equation of state is not harmonic (constant or even zero in the ideal case). However, the pressure field of TGV must be harmonic (according to equation 3.1). With the previous improvement, the relative error has been reduced considerably (figure 3.4) and all previously mentioned improvements have also been observed again. The following subsection contains the results and discussion for the improved pressure.

3.1.3 Improvement of results with improving pressure's accuracy

As mentioned before, equation 3.1 is added to the obtained pressure from the equation of state to improve the results further. For this case, the pressure-ratio constant is chosen to be 1.0 but, the reference density is chosen to be the minimum density at each time step. Because the variation of density throughout the domain and time steps becomes very small, the pressure obtained from the equation of state for this case is almost zero. Therefore, with the addition of equation 3.1, the obtained pressure field is corrected to be the same as the analytical solution. This improves the relative error considerably as the figure 3.4 shows. The error has become acceptable concerning the corresponding error for this case in figure 5b of [Adami et al.(2013)].

As discussed above and also in the previous subsection, the added analytical pressure also kept the previously achieved improvements untouched. Therefore, the corresponding figures are shown only for one of the cases.

First, concerning the particle configuration, it was noted that there will be no alignment of particles along their stagnation lines using the improvements in this and the previous subsection. The snapshot of particle configurations at $t = 0.0$, $t = 0.2$, $t = 0.4$ and $t = 0.6$ s are shown in figure 3.5. By comparing this figure with figures 4 and 6 of [Adami et al.(2013)], the elimination of the said problem is visible.

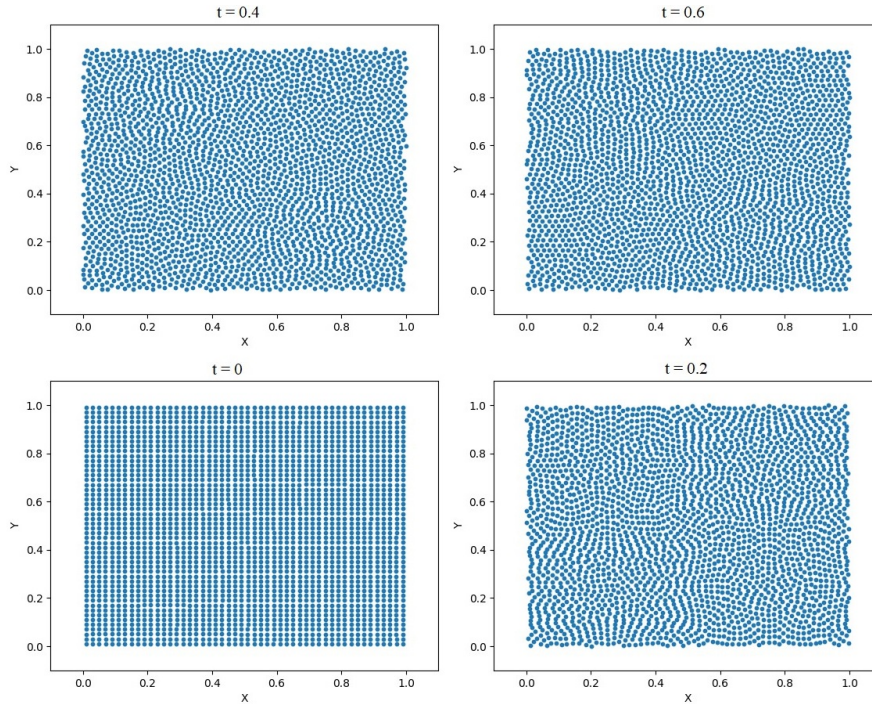


Figure 3.5 This figure contains the snapshots of particle configurations at $t = 0$, $t = 0.2$, $t = 0.4$ and $t = 0.6$ s for the case discussed in subsection 3.1.2. It is evident that no alignment along stagnation lines occurs while the initial particle configuration is uniform and Cartesian. However, this is not the case for the figure 4 of [Adami et al.(2013)].

The weakly compressible assumption (equation 2.1) is also held at least for these two cases throughout the simulation. Figure 3.6 shows the density distribution of the previous case at $t = 2.4$ s, which has almost one percent of variation at the end of the simulation. Figure 7b of [Adami et al.(2013)] also shows the same situation but, not for an initial Cartesian, uniform particle configuration.

Figure 7a of [Adami et al.(2013)] also shows the preservation of velocity field's form for TGV but, again, only for the case that uses the particle configuration of a pre-run as an initial particle configuration. However, here, the preservation of velocity field's form is seen even for the case with initial Cartesian particle configuration. This is shown in figure 3.7.

3.2 Development of a code in JAX for exact calculation of the energy spectrum for velocities on regular and irregular grids

The purpose here is to have a code in JAX that calculates energy spectrum exactly for simulations done on regular and irregular grids (or the Eulerian and Lagrangian formulations). The exact calculation of energy spectrum is crucial in analysing the accuracy of turbulence simulations.

As explained in chapter 2, the exact calculation is important specifically for large wavenumbers. This importance stems from two facts. One is that the damping of energy in turbulence happens in high wavenumbers (smallest vortices). Therefore, the accuracy of the whole turbulence simulation is based on the correctness of the energy spectrum for all wavenumbers. The other fact is that calculating energy spectrum can become erroneous specifically for large wavenumbers. Therefore, special attention must be paid to the calculation of the energy spectrum itself so that no errors are introduced to the data obtained from the simulations.

Based on the literature, as discussed in chapter 1, the energy spectrum for velocities obtained from a regular grid is exact, if it is obtained correctly through Fast Fourier Transform (FFT). The most accurate formulation that exists for velocities on an irregular grid (irregular shape or spacing) is proposed by [Shi et al.(2013)]. The formulation is the second-order Moving Least Squares (MLS).

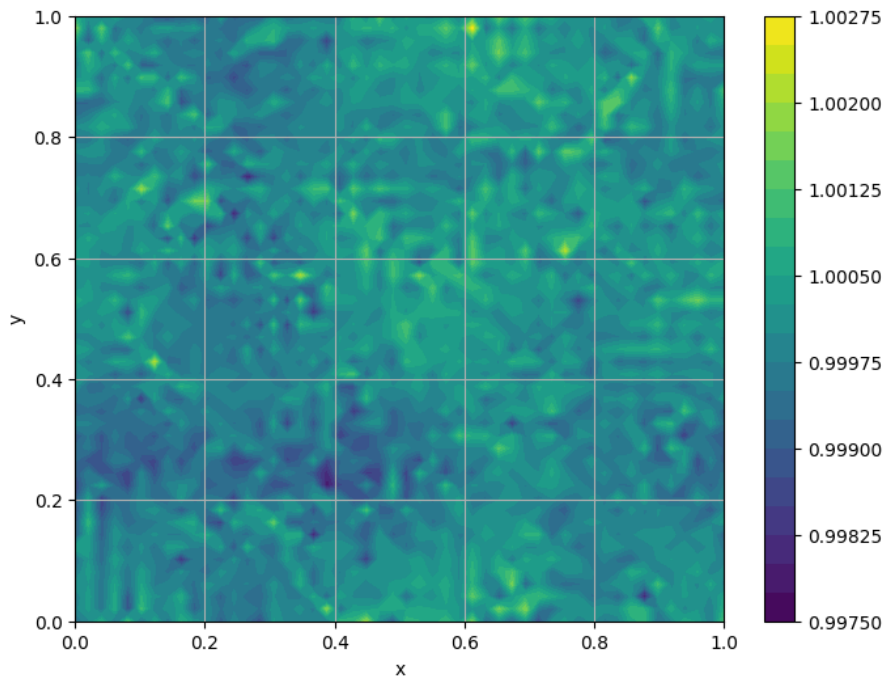


Figure 3.6 The density distribution of a case with the improvements of subsection 3.1.2 at $t = 2.4$ s. The distribution shows almost one percent of variation at this time (the end of the simulation).

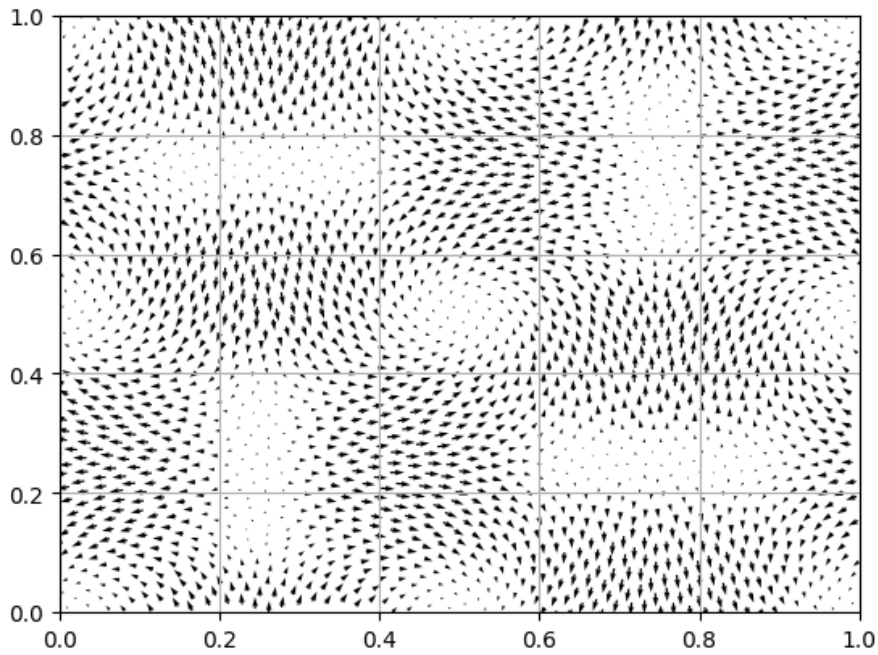


Figure 3.7 The velocity field of the case presented in subsection 3.1.3 at $t = 1.0$ s. This velocity field and the velocity fields at the time steps before, all, have the same form as the initial velocity field of TGV (figure 3.1).

As discussed in chapter 2, second-order MLS is an interpolation technique that interpolates velocities of an irregular grid on a regular grid with the least introduced error. Therefore, an FFT applied to the interpolated velocity can lead to the energy spectrum with the least introduced error.

Based on the information above, a code that can perform FFT and MLS formulations correctly, can calculate the energy spectrum exactly throughout whole wavenumbers.

In the following subsection, subsection 3.2.1, the results for the energy-spectrum calculation with FFT with the developed JAX code are presented and validated with data from [Shi et al.(2013)].

The subsequent subsection contains the results and validation of second-order MLS interpolation for the developed JAX code.

3.2.1 Calculation of energy spectrum with FFT and validation with the results from [Shi et al.(2013)]

In order to validate the calculation of energy spectrum with FFT, figure 8b of [Shi et al.(2013)] is recreated. [Shi et al.(2013)] uses the initial velocity field proposed by [Rogallo(1981)] to calculate the energy spectrum.

[Rogallo(1981)] has proposed a fourier-transformed velocity field that can result in a specific energy spectrum. In other words, first, the desired energy-spectrum function is chosen for the field. Then, based on that energy spectrum, the transformed-velocity field is calculated. If the energy spectrum of that transformed-velocity field is calculated again, it should result in exactly the chosen energy spectrum. Therefore, it is a good choice to benchmark energy-spectrum calculation with.

In figure 8b of [Shi et al.(2013)], energy-spectrum function of $E(k) = k^{-5/3}$ is chosen. Then, based on the previous function, the transformed-velocity field is calculated. In [Rogallo(1981)] and similarly, in the appendix of [Shi et al.(2013)], the details of the calculation of the transformed-velocity field from a desired energy spectrum are explained. Here, only the formulations are presented but, with a small correction that is needed due to a typo that could be found by following the proof.

The transformed-velocity field ($\mathbf{V}(\mathbf{k})$) to achieve the desired energy-spectrum function of $E(k)$ is:

$$\mathbf{V}(\mathbf{k}) = \left(\frac{\alpha k k_2 + \beta k_1 k_3}{k(k_1^2 + k_2^2)^{1/2}} \right) \mathbf{e}_1 + \left(\frac{\beta k_2 k_3 - \alpha k k_1}{k(k_1^2 + k_2^2)^{1/2}} \right) \mathbf{e}_2 + \left(\frac{\beta(k_1^2 + k_2^2)^{1/2}}{k} \right) \mathbf{e}_3. \quad (3.5)$$

In the above equation, $\mathbf{k} = (k_1, k_2, k_3)$ and $k = |\mathbf{k}|$. k_1 , k_2 and k_3 are wavenumbers in direction of \mathbf{e}_1 , \mathbf{e}_2 and \mathbf{e}_3 respectively. \mathbf{e}_1 , \mathbf{e}_2 and \mathbf{e}_3 are the unit vectors of an Eulerian space (ie. the unit vectors are perpendicular pairwise). α and β are calculated as follows:

$$\alpha = \left(\frac{E(k)}{4\pi k^2} \right)^{1/2} e^{i\theta_1} \cos(\phi) \quad (3.6)$$

$$\beta = \left(\frac{E(k)}{4\pi k^2} \right)^{1/2} e^{i\theta_2} \sin(\phi). \quad (3.7)$$

In the above, θ_1 , θ_2 and ϕ are uniformly distributed random numbers on the interval $(0, 2\pi)$. i is $\sqrt{-1}$.

As discussed before, first, the desired function for $E(k)$ is selected. Then, α and β are calculated and, finally, $\mathbf{V}(\mathbf{k})$ is known.

As mentioned earlier, $\mathbf{V}(\mathbf{k})$ is the Fourier transformed of a velocity field. Therefore, with the conjugation of $\mathbf{V}(\mathbf{k})$, energy spectrum ($E(k)$) can be calculated, which is expected to be exactly the earlier chosen function.

If the above routine is followed by the JAX code, it shows that FFT calculation of energy spectrum in the code is correct because except for the Fourier transformation of the velocity field, all other steps are repeated exactly on any velocity field in order to obtain the energy spectrum. The Fourier transformation of velocity field in JAX can be done by internal libraries and expected to be accurate (while done numerically). The only point that must be considered is that the kind of transformation in JAX must be orthogonal. This has also been implicitly mentioned in [Shi et al.(2013)]. Orthogonality of Fourier transformation insures its

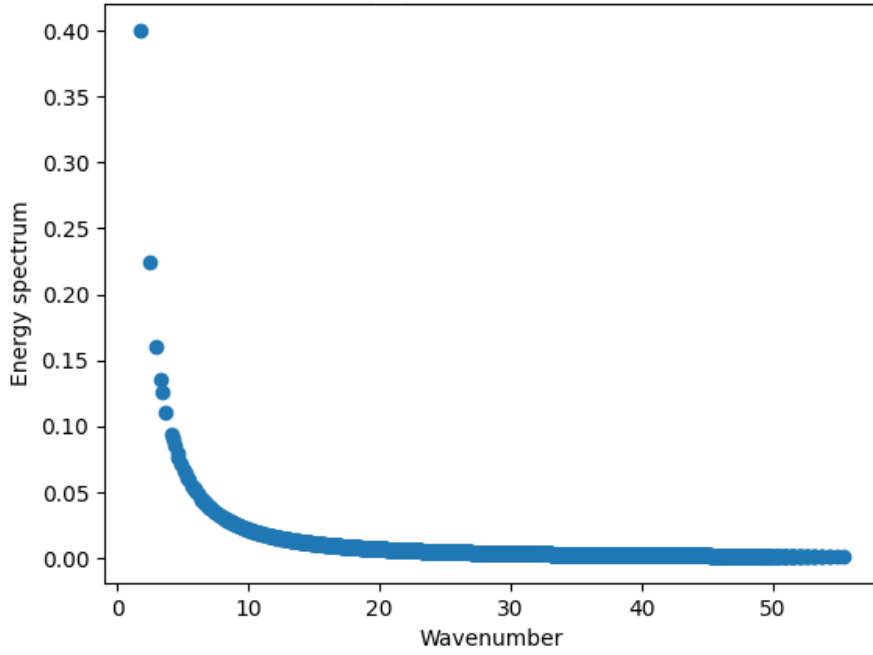


Figure 3.8 Calculated energy spectrum of $E(k) = k^{-5/3}$ versus k in a normal plot.

reversibility. Therefore, FFT of data that are distributed on points with the same distance (regular grid) will have exact results and no further error will be introduced.

The mathematical reasoning of the above is as follows:

if $\mathbf{V}(\mathbf{K})$ is the fourier transformation of the velocity field, $\mathbf{v}(\mathbf{r})$, the matrix form of transformation will be:

$$\mathbf{V} = B\mathbf{v} \quad (3.8)$$

where B is transformation matrix of $e^{-i\mathbf{r}\cdot\mathbf{k}}$. The inverse of the previous can be written as:

$$\mathbf{v} = \frac{1}{N}B^H\mathbf{V}. \quad (3.9)$$

In the above, N is the number of data and B^H is the Hermitian transpose or conjugate transpose of matrix B. Based on the above, to observe the reversibility of Fourier transformation, $\frac{1}{N}(BB^H) = I$ must be true to result in $\mathbf{V} = \frac{1}{N}(BB^H)\mathbf{V}$ (I is the identity matrix.). Therefore, Fourier transformation setting of JAX should be set to "ortho".

The calculated energy spectrum from $\mathbf{V}(\mathbf{k})$ (equation 3.5) and for $E(k) = k^{-5/3}$ is shown in figure 3.8. The logarithmic plot for both axes are in figure 3.9. The relative error between the calculated energy spectrum and $E(k) = k^{-5/3}$ with regard to $E(k) = k^{-5/3}$ in percent is in figure 3.10. The extreme small errors in figure 3.10 shows the high accuracy of FFT routine for all wavenumbers.

In order to make sure that the accurate results in figure 3.10 are not accidental, the procedure above is repeated for $E(k) = k^{-5}$. -5 is a steeper slope. Hence, it is expected to reveal deviations between the analytical and calculated energy spectrum better.

Again, the results prove the high accuracy of energy-spectrum calculation. The calculated energy spectrum from $\mathbf{V}(\mathbf{k})$ and for $E(k) = k^{-5}$ is shown in figure 3.11. The logarithmic plot for both axes are in figure 3.12. The relative error between the calculated energy spectrum and $E(k) = k^{-5}$ with regard to $E(k) = k^{-5}$ in percent is in figure 3.13. The extreme small errors in figure 3.13 prove the high accuracy of FFT routine for all wavenumbers and also for sharp variations of energy spectrum throughout different wavenumbers.

Based on the results above, it is safe to conclude that the FFT routine for 3D velocity fields is validated (3D because the transformed-velocity field is in 3D).

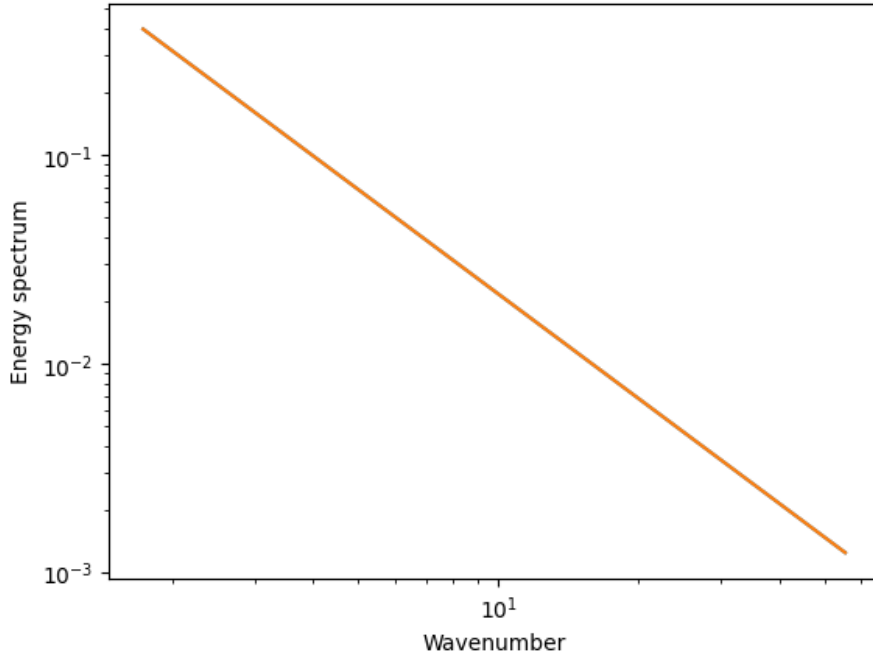


Figure 3.9 Energy spectrum of $E(k) = k^{-5/3}$ versus k in a log-log plot. There are two diagrams in this figure. One is for $E(k) = k^{-5/3}$ and the other is the calculated energy spectrum from the corresponding transformed-velocity field. Because the errors between the two are extremely small, one diagram can only be seen. In order to avoid confusion, no legend is added. The relative errors in percent between the two diagrams are shown in figure 3.10.

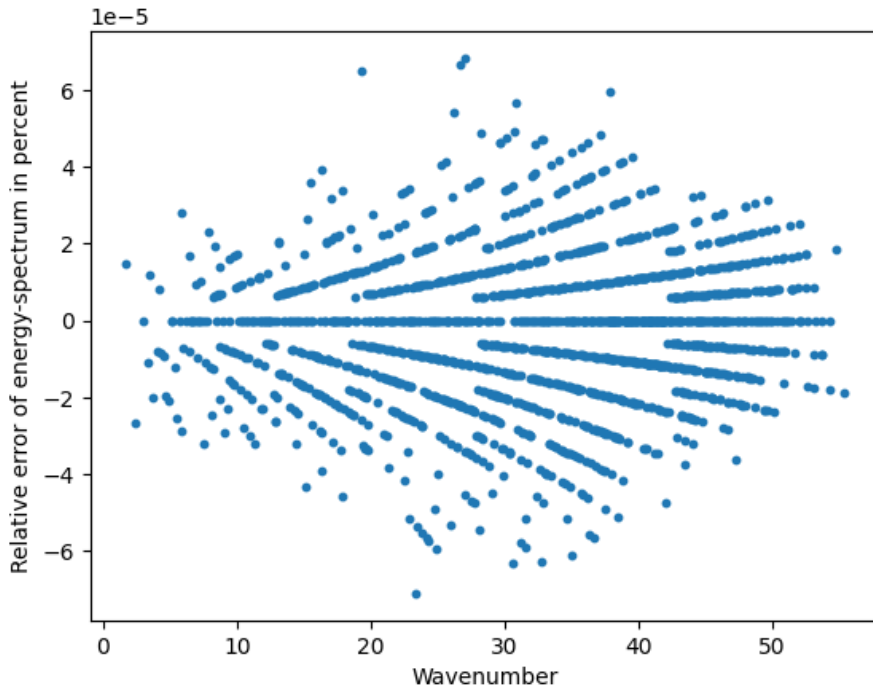


Figure 3.10 the relative error between energy spectrum of $E(k) = k^{-5/3}$ and the calculated energy spectrum with FFT from the transformed-velocity field in percent versus k . The error is relative to $E(k) = k^{-5/3}$. The extremely small errors show very accurate results for all wavenumbers.

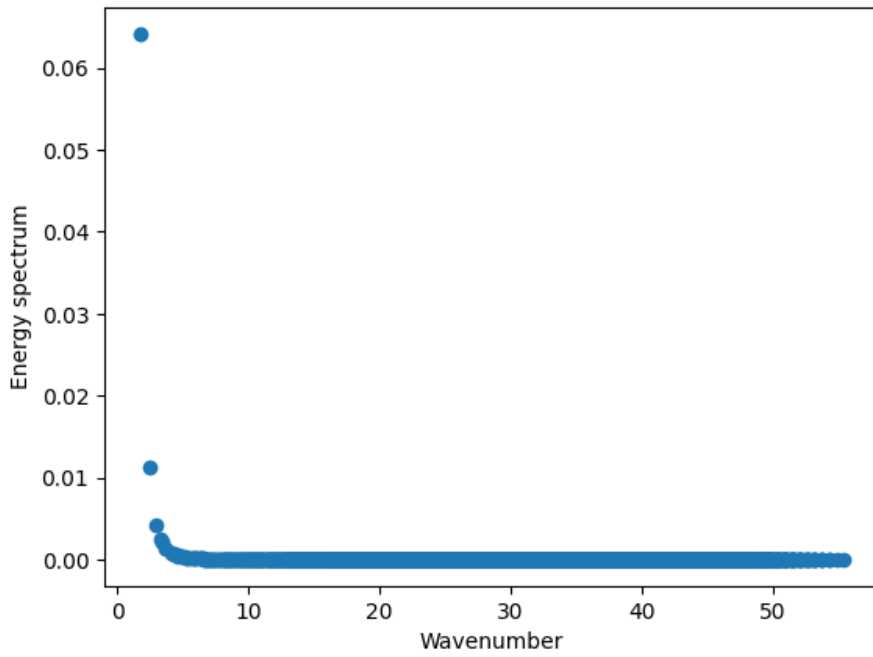


Figure 3.11 Calculated energy spectrum of $E(k) = k^{-5}$ versus k in a normal plot.

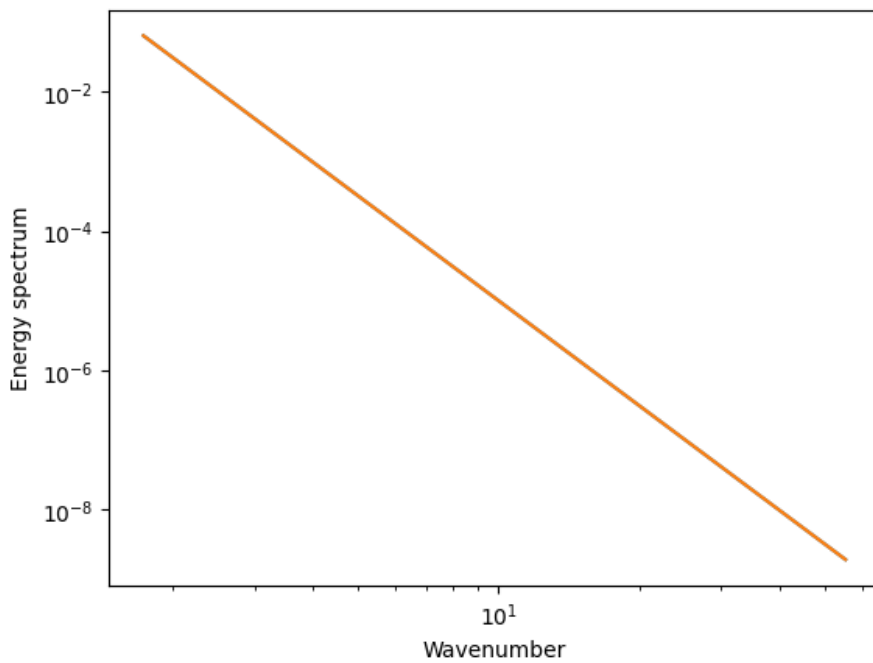


Figure 3.12 Energy spectrum of $E(k) = k^{-5}$ versus k in a log-log plot. There are two diagrams in this figure. One is for $E(k) = k^{-5}$ and the other is the calculated energy spectrum from the corresponding transformed-velocity field. Because the errors between the two are extremely small, one diagram can only be seen. In order to avoid confusion, no legend is added. The relative errors in percent between the two diagrams are shown in figure 3.13.

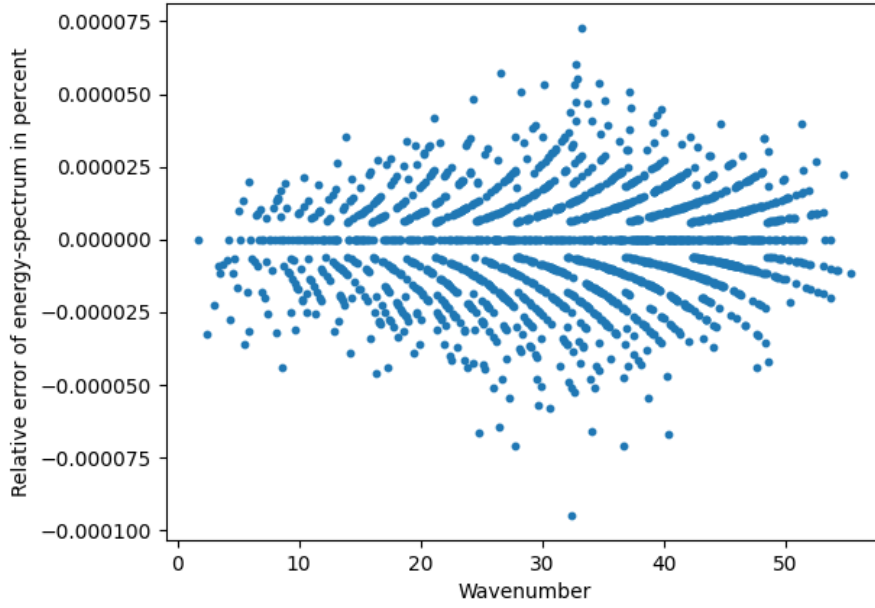


Figure 3.13 the relative error between energy spectrum of $E(k) = k^{-5}$ and the calculated energy spectrum with FFT from the transformed-velocity field in percent versus k . The error is relative to $E(k) = k^{-5}$. The extremely small errors show very accurate results for all wavenumbers.

3.2.2 Calculation of energy spectrum with MLS

FFT calculation of energy spectrum in the previous section is exact for velocity field that is distributed on a regular grid. For an irregular grid, either irregular shape of the grid or irregular distance between the grid vertexes, previous FFT calculation is not as accurate as before.

According to the literature review in chapter 1, the current most accurate routine to calculate energy spectrum for irregular grids, including SPH, is proposed by [Shi et al.(2013)]. The routine is the second-order MLS that is discussed in chapter 2.

Second-order MLS interpolates the velocity field as accurate as possible on a regular grid. From that point, energy spectrum can be calculated with FFT from the interpolated velocity field.

Here, the correctness of developed 2D MLS in JAX is validated. Because in the previous section, 3D FFT is validated, first, the correctness of FFT for 2D is shown.

In [Rogallo(1981)] and [Shi et al.(2013)], there is no transformed-velocity field for 2D to check the correctness of FFT. Therefore, the transformed-velocity for 3D (equation 3.5) is changed to 2D. The resulted 2D transformed-velocity field is:

$$\mathbf{V}(\mathbf{k}) = \left(\frac{\alpha k k_2}{k(k_1^2 + k_2^2)^{1/2}} \right) \mathbf{e}_1 + \left(\frac{-\alpha k k_1}{k(k_1^2 + k_2^2)^{1/2}} \right) \mathbf{e}_2 \quad (3.10)$$

where $\mathbf{k} = (k_1, k_2)$ and $k = |\mathbf{k}|$. Here, α is for 2D. Consequently, $\alpha = \left(\frac{E(k)}{2\pi k} \right)^{1/2} e^{i\theta_1} \cos(\phi)$. The parameters in the calculation of α are calculated the same as before.

It is not expected that the above \mathbf{V} results exactly in the selected energy-spectrum function. However, a close result is expected. Figures 3.14 and 3.15 show the calculated energy spectrum for 2D. Figure 3.15 shows that the changed routine from the validated 3D FFT to 2D FFT is correct (Other tests have also been carried out, eg. applying a constant \mathbf{V} , and all led to the correctness of 2D FFT. In order to shorten the discussion, they are not discussed here.). Therefore, 2D FFT in JAX is correct.

The 2D FFT is developed because in order to calculate the energy spectrum for 2D MLS, the routine is used. Hence, it should have been validated. However, because MLS is an interpolation technique, MLS can also be validated if the interpolated values can be known beforehand so that the interpolated and expected results can be compared against each other. Based on the previous idea, as a validation for

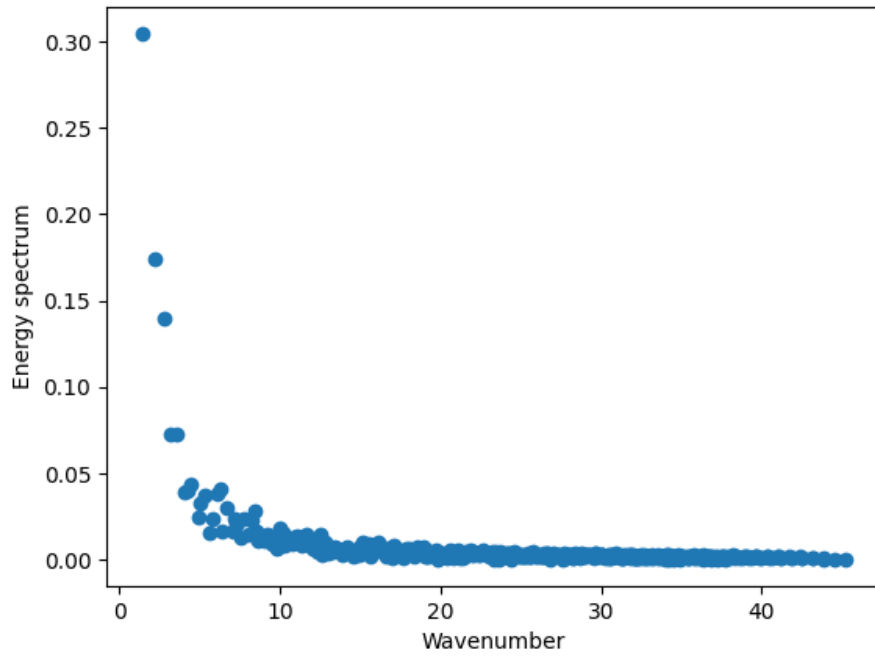


Figure 3.14 Calculated energy spectrum for a 2D domain with the transformed-velocity field of equation 3.10. The axes have normal scale.

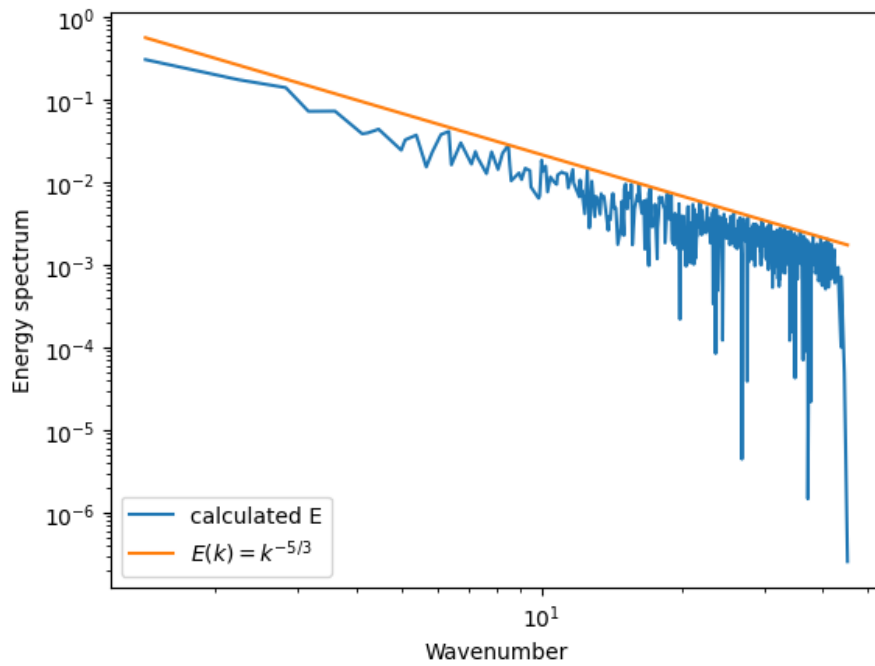


Figure 3.15 Calculated and expected energy spectra are shown in this figure. The expected energy spectrum is $E(k) = k^{-5/3}$ because the domain has the corresponding transformed-velocity field (equation 3.10).

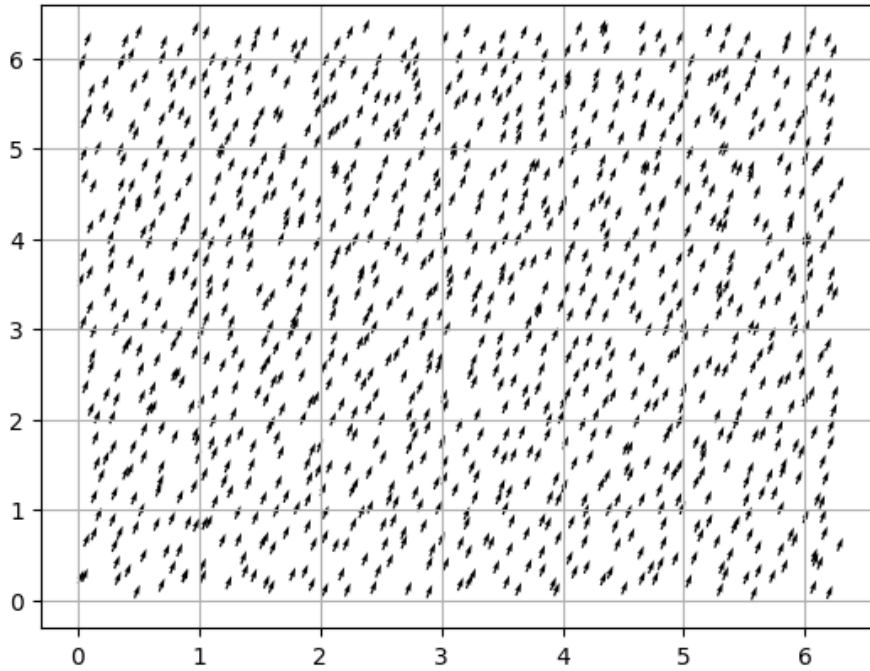


Figure 3.16 Velocity field of $v = (2.0, 5.0)$ on an irregularly, randomly placed particles.

the developed second-order MLS, a constant velocity field is applied to a 2D domain but, on an irregular grid. Because for all particles, the velocity is constant, it is expected that the interpolated values by MLS become the same as the values on the irregular grid. Figures 3.16 and 3.17 show the results. Figure 3.16 is the constant velocity field of $v = (2.0, 5.0)$ for a randomly, irregularly placed particles. The interpolated velocity field is in figure 3.17.

Although figures 3.16 and 3.17 can, to a degree, show the correctness of the developed MLS routine in JAX, in order to completely show the correctness of the developed code, MLS is tested for a more complex velocity field and this time, even the energy spectrum is calculated as well. Because the correctness of developed code for 2D FFT is validated, deviations can be blamed on the code for the MLS routine. However, the results once again prove the correctness of the MLS code. The detailed explanation of the previous proof is as follows:

First, a suitable velocity field must have been chosen that was complex. In addition, while it was applied to an irregular grid, its interpolated values on the regular must have been known to be compared with the results from MLS. Based on the previous features, the initial velocity field of a 2D TGV (equations 3.2 and 3.3) was chosen. For equations 3.2 and 3.3, the 2D domain is chosen to be $(6\pi, 6\pi)$, which is the chosen domain in [Shi et al.(2013)]. The resulted energy spectra are in figure 3.18. Resemblance of energy spectrum from the regular and irregular grid in figure 3.18 shows the correctness of the code that calculates second-order MLS.

In the next chapter, the conclusions of this research are presented.

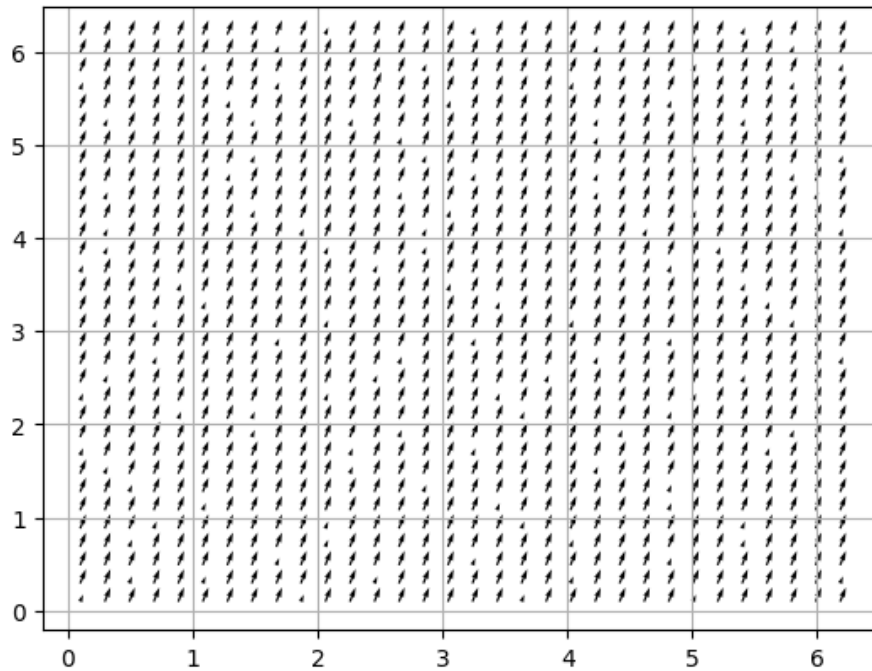


Figure 3.17 Interpolated velocity field of figure 3.16 by second-order MLS on a regular grid. For most of the points, the expected field of $v = (2.0, 5.0)$ is achieved. However, on some points, there are some errors. The errors are due to the condition of the coefficient matrix of the system of equations solved for MLS.

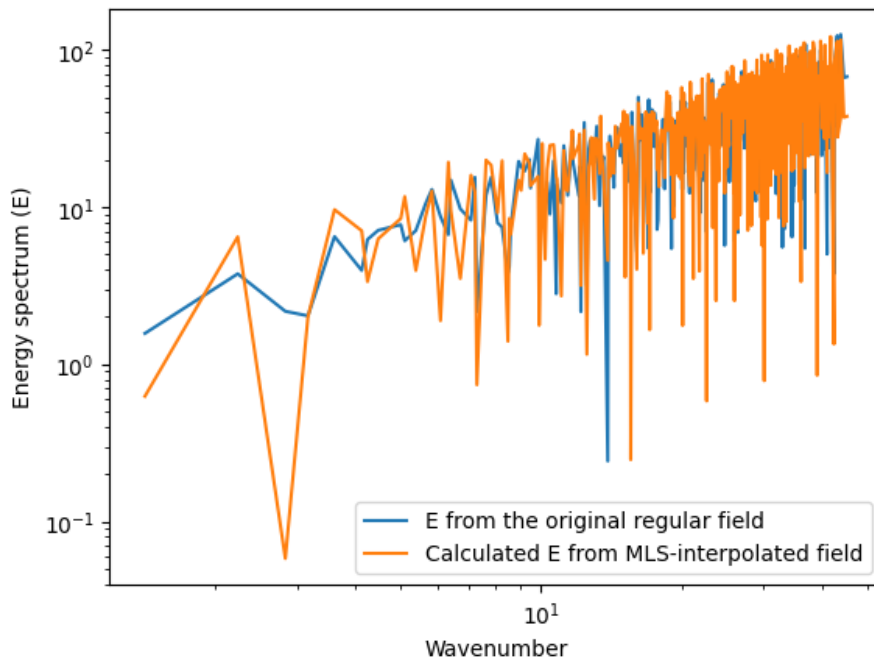


Figure 3.18 This figure contains two diagrams. Blue line is the energy spectrum that is calculated with the validated 2D FFT and from a 2D TGV velocity field on a regular grid. Based on the equations 3.2 and 3.3, the corresponding velocity field for a randomly, irregularly placed particles is formed. Then, using the second-order MLS, the velocity field is interpolated (which should result in the previous TGV field). Then, the energy spectrum is calculated using FFT, which has been shown in the figure with an orange line. Because the blue and orange line resemble each other, it is safe to conclude that the code for the MLS interpolation of a 2D field is correct.

4 Conclusions

This research can be divided into two parts. One part is for the development of a JAX-SPH code and the next is the exact calculation of energy spectrum. The conclusions of each part are presented in its corresponding section.

4.1 Conclusions of part 1, development of a JAX-SPH code and validation of its results with the results from [Adami et al.(2013)]

In part 1, the main goal was to develop the first working instance of a differentiable CFD solver based on a Lagrangian formulation. According to the literature review done during this research, there was no differentiable CFD solver that was based on Lagrangian formulations (particularised domain). Therefore, the research's main idea was novel.

Because recently, some differentiable CFD solvers had been developed in JAX successfully (e.g. [Kochkov et al.(2021)], [Bezgin et al.(2023)]), it was decided that this research would be performed entirely in JAX. The Lagrangian formulation for the first part was chosen to be Smoothed Particle Hydrodynamics because a solid history of research in that field exists at TUM. Exactly due to the previous reason, article of [Adami et al.(2013)], and in particular the Taylor-Green Vortex (TGV) problem, was selected to be the basis for validation. TGV is a problem that has been investigated both analytically and numerically for more than 50 years. The information also spans the whole regimes of laminar, transition and turbulence. Therefore, results from the developed solver were compared with the results from the TGV problem.

Because the results from the SPH simulations depend slightly on some numerical coefficients (that do not necessarily result from physical principles), some trials were necessary to select the best set of coefficients that could reproduce the results from [Adami et al.(2013)] exactly. The trials were successful and the solver's results were validated. During those trials, some techniques for the improvement of the formulation of [Adami et al.(2013)] were also developed.

First technique is to set the reference density equal to the minimum density in the domain at each time step. Consequently, pressure field will maintain its positive value in the whole domain at all time steps. Since the previous technique had not been introduced before, to the best of our knowledge, it was coined as the "adaptive-reference-density" technique. The term "adaptive" emphasises the change of reference density based on the variations in the density field at each time step as opposed to a constant reference density for all time steps, which is normally used by other researchers.

The second developed technique was to choose the reference density as the minimum density of each time step but, this time, divided by a constant. The constant was coined "pressure-ratio constant". With the help of this constant, the impact of different pressures introduced in [Adami et al.(2013)] could be controlled. In [Adami et al.(2013)], there are three different pressures: momentum, reference and background pressure. Momentum pressure is the famous general pressure. Reference pressure is specific to SPH simulations, which is calculated with the reference density and the speed of sound and is mostly a numerical coefficient.

[Adami et al.(2013)] introduced the background pressure for the first time. Its main goal is to prevent particles from clustering and particle displacement in each time step is dependent on that. However, the momentum velocity is under the influence of momentum pressure and particles are supposed to move based on the momentum velocity and not background pressure (and transport velocity).

Consequently, the balance among those pressures are important. According to the equations, pressure-ratio constant of 2.0 can create the best results because it makes the three pressures almost equal.

With the two techniques above, the results obtained from the JAX solver can be better than the ones depicted in [Adami et al.(2013)]. [Adami et al.(2013)] has reported the alignment of particles along their stagnation lines at the beginning of their simulations. It seems this will also lead to some other errors. Consequently, its prevention can improve the results. [Adami et al.(2013)] prevents the alignment of particles by re-running the same simulation with the particles being initialised with a pre-run's last particle configuration and not with a uniformly placed, Cartesian configuration. The results from the JAX code do not show any alignment of particles. Therefore, they show a uniform distribution of density field (less errors).

The only aspect that the two techniques above could not improve, was the relative error of the decay of maximum velocity of TGV. However, results could be improved further by adding the analytical solution of the pressure of the TGV to the equation of state (which ideally results in a zero pressure with pressure-ratio constant of 1.0). With the previous improvement, the relative error can also be controlled and be held at an acceptable level concerning the errors reported by [Adami et al.(2013)].

Therefore, the results from JAX code not only became validated but also, they did not show any alignment of particles along their stagnation lines. Hence, a more uniform density field was achieved and the form of TGV's velocity field was mostly preserved. The innovation here is that the previous results could be achieved with placing particles initially on a uniform, Cartesian grid and without using a pre-run.

4.2 Conclusions of part 2, development of a JAX code that calculates energy spectrum exactly for velocities from regular or irregular grids

In order to assess the accuracy of turbulence simulations, comparisons made between energy spectra can be very informative or even crucial. Therefore, an exact calculation of energy spectrum is crucial for turbulence simulations.

If the velocities are distributed on a regular, uniformly placed grid, the energy spectrum can be calculated with Fast Fourier Transformation (FFT) exactly. If velocities are distributed on an irregular grid, the current most accurate method to calculate the energy spectrum is second-order Moving Least Squares (MLS) method proposed by [Shi et al.(2013)].

The developed code in JAX exactly calculates the energy spectrum for velocities on a regular grid with FFT. For irregular grids and second-order MLS, the results showed accurate energy-spectrum calculation as expected. Errors existed in the results from MLS but, they were related to the condition of the system of equations (condition of the coefficient matrix) that was solved for each point.

Bibliography

- [Adami et al.(2013)] S. Adami, X.Y. Hu, and N.A. Adams. 2013. A transport-velocity formulation for smoothed particle hydrodynamics. *J. Comput. Phys.* 241 (2013), 292–307. <https://doi.org/10.1016/j.jcp.2013.01.043>
- [Bezgin et al.(2023)] Deniz A. Bezgin, Aaron B. Buhendwa, and Nikolaus A. Adams. 2023. JAX-Fluids: A fully-differentiable high-order computational fluid dynamics solver for compressible two-phase flows. *Computer Physics Communications* 282 (2023), 108527. <https://doi.org/10.1016/j.cpc.2022.108527>
- [Kochkov et al.(2021)] Dmitrii Kochkov, Jamie A. Smith, Ayaa Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. 2021. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences* 118, 21 (2021), e2101784118. <https://doi.org/10.1073/pnas.2101784118> arXiv:<https://www.pnas.org/doi/pdf/10.1073/pnas.2101784118>
- [Rogallo(1981)] R. S. Rogallo. 1981. Numerical experiments in homogeneous turbulence. *NASA TM* (1981), 81315.
- [Shi et al.(2013)] Yilei Shi, Xiao Xiang Zhu, Marco Ellero, and Nikolaus A. Adams. 2013. Analysis of interpolation schemes for the accurate estimation of energy spectrum in Lagrangian methods. *Computers and Fluids* 82 (2013), 122–131. <https://doi.org/10.1016/j.compfluid.2013.05.003>
- [Taylor and Green(1937)] G. I. Taylor and A. E. Green. 1937. Mechanism of the Production of Small Eddies from Large Ones. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences* 158, 895 (1937), 499–521. <http://www.jstor.org/stable/96892>
- [Xenakis et al.(2015)] A.M. Xenakis, S.J. Lind, P.K. Stansby, and B.D. Rogers. 2015. An incompressible SPH scheme with improved pressure predictions for free-surface generalised Newtonian flows. *Journal of Non-Newtonian Fluid Mechanics* 218 (2015), 1–15. <https://doi.org/10.1016/j.jnnfm.2015.01.006>