

# 3D Taylor-Green Vortex Comparison

Paul Bartholomew

March 1, 2019

## Abstract

As part of the verification of `Xcompact3D` we simulate the Taylor-Green vortex and compare with results from a reference 6<sup>th</sup> order compact finite difference code provided by Eric Lamballais.

## Contents

<b>1 Introduction</b>	<b>1</b>
<b>2 Computational setup</b>	<b>1</b>
2.1 Discretisation . . . . .	1
2.2 Runtime parameters . . . . .	2
<b>3 Comparison of results</b>	<b>2</b>

## 1 Introduction

As a canonical test case, the Taylor-Green vortex provides a check that the time integration of the Navier-Stokes is working correctly. The Taylor-Green vortex is initialised as follows

$$\mathbf{u} = \begin{cases} U \sin(x/\pi) \cos(y/\pi) \cos(z/\pi) \\ -U \cos(x/\pi) \sin(y/\pi) \cos(z/\pi) \\ 0 \end{cases} \quad (1)$$

in the  $-\pi \leq \mathbf{x} \leq \pi$  periodic box. To save computational effort, the symmetries inherent in the flow field are exploited to simulate only the impermeable sub-domain  $0 \leq \mathbf{x} \leq \pi$ .

## 2 Computational setup

### 2.1 Discretisation

To ensure comparability of the results we must first ensure the same schemes are being used - the pertinent variables are `fpi2` and `ailcaix6` (and `y` and `z`) which should be set to  $48/7/\pi^2$  and 0.461658 in both codes.

## 2.2 Runtime parameters

The runtime parameters pertinent to the simulation are given in table 1.

Table 1: Runtime parameters for Taylor-Green Vortex simulations.

Parameter	Value	Notes
<code>xlx</code>	3.14159265358979	<code>ylx</code> and <code>zlx</code> the same
<code>nx</code>	65	<code>ny</code> and <code>nz</code> the same equivalent to a $129^3$ domain
<code>nclx</code>	1	<code>nclx</code> and <code>nclz</code> the same corresponds to free-slip
<code>dt</code>	0.001	
Time scheme	RK3	
<code>ilast</code>	20,000	
Output frequency	1,000	Stores snapshots every 1,000 steps

## 3 Comparison of results

The main statistics of interest for comparison purposes are the kinetic energy and enstrophy, defined as

$$k = \frac{1}{2} \int_{\Omega} \mathbf{u}^2 dV , \quad (2)$$

and

$$\varepsilon = \int_{\Omega} |\boldsymbol{\omega}|^2 dV , \quad (3)$$

where

$$\boldsymbol{\omega} = \nabla \times \mathbf{u} , \quad (4)$$

is the vorticity.

The codes compute these statistics online, here a `python` script has been developed to plot them for comparison. It expects that the data are located in `./x3d/time_evol.dat` and `./e3d/time_evol.dat` with the following format (comment lines beginning with a `#` and additional columns are ignored)

<code>#</code>	<code>TIME</code>	<code>KE</code>	<code>DISS</code>	<code>ENST</code>	<code>DISS2</code>
	<code>t1</code>	<code>k1</code>	<code>d1</code>	<code>e1</code>	<code>D1</code>
	<code>t2</code>	<code>k2</code>	<code>d2</code>	<code>e2</code>	<code>D2</code>
	<code>...</code>				
	<code>tn</code>	<code>kn</code>	<code>dn</code>	<code>en</code>	<code>Dn</code>

which can be read by following `python` snippet given in listing 1.

```

def read_stats(filename):
    t = []
    enst = []
    ke = []

    with open(filename, "r") as data:
        for row in data:
            if not (row[0]=="#"):
                words = row.split()
                t.append(float(words[0]))
                enst.append(float(words[1]))
                ke.append(float(words[3]))

    return t, enst, ke

```

Listing 1: Python code to read statistics for TGV case.

The data are plotted using `matplotlib` in listing 2.

```

def plot_stats(x3d_t, x3d_dat, x3d_lab, e3d_t, e3d_dat, e3d_lab,
              xlab, ylab, outfile, figsize=(5.0, 3.5)):

    plt.figure(figsize=figsize)

    plt.plot(x3d_t, x3d_dat, label=x3d_lab)
    plt.plot(e3d_t, e3d_dat, label=e3d_lab)

    plt.xlabel(xlab)
    plt.ylabel(ylab)
    plt.legend()

    plt.savefig(outfile, bbox_inches="tight")
    plt.close()

```

Listing 2: Python code to plot comparison of `Xcompact3D` and Eric's reference code.

And finally, the following script (`plot_tgv.py`) plots the data in *fig. 1* and *fig. 2*.

```

import matplotlib.pyplot as plt

<<src:read-stats.py>>
<<src:plot-stats.py>>

x3d_t, x3d_enst, x3d_ke = read_stats("./x3d/time_evol.dat")
e3d_t, e3d_enst, e3d_ke = read_stats("./e3d/time_evol.dat")

# Made an error in X3D, all time shifted by -dt=1e-2
for i in range(len(x3d_t)):
    x3d_t[i] += 1.0e-2

plot_stats(x3d_t, x3d_enst, "X3D",
           e3d_t, e3d_enst, "Eric",
           r"$t$", r"$\varepsilon$",
           "tgv_enstrophy.eps")
plot_stats(x3d_t, x3d_ke, "X3D",
           e3d_t, e3d_ke, "Eric",
           r"$t$", r"$k$",

```

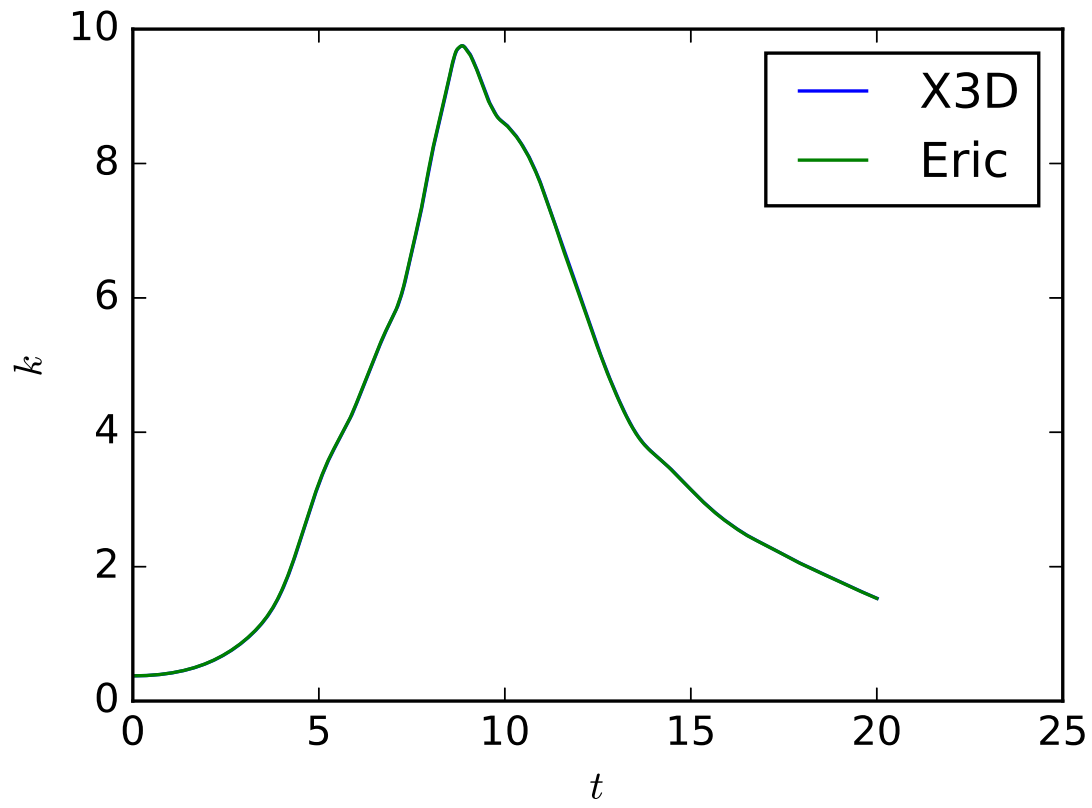


Figure 1: Comparison of kinetic energy

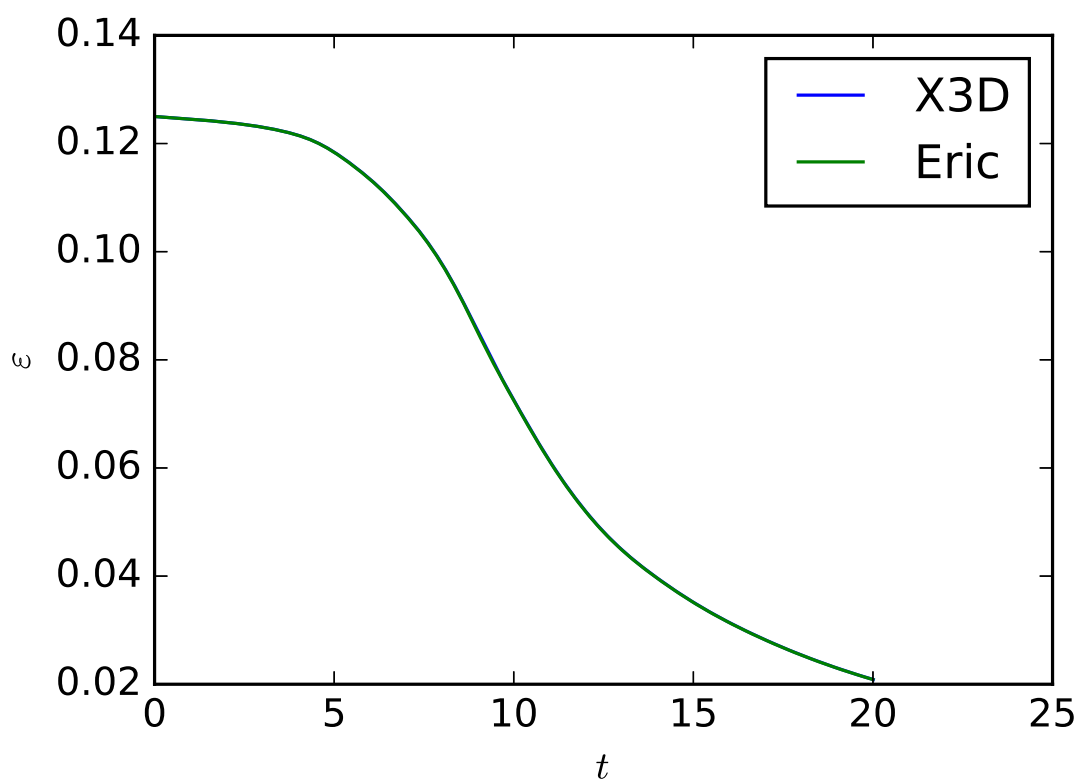


Figure 2: Comparison of enstrophy