Available online at www.sciencedirect.com

ScienceDirect

Journal of Computational Physics 227 (2007) 12–35

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Numerical comparison of Riemann solvers for astrophysical hydrodynamics

Christian Klingenberg [a], Wolfram Schmidt [b], Knut Waagan [c,*]

[a] Department of Mathematics, Würzburg University, Am Hubland, 97074 Würzburg, Germany
[b] Department of Theoretical Physics, Würzburg University, Am Hubland, 97074 Würzburg, Germany
[c] Center of Mathematics for Applications, P.O. Box 1053, Blindern, NO-0316 Oslo, Norway

Received 6 December 2006; received in revised form 20 May 2007; accepted 5 July 2007
Available online 17 August 2007

Abstract

The idea of this work is to compare a new positive and entropy stable approximate Riemann solver by Francois Bouchut with a state-of the-art algorithm for astrophysical fluid dynamics. We implemented the new Riemann solver into an astrophysical PPM-code, the Prometheus code, and also made a version with a different, more theoretically grounded higher order algorithm than PPM. We present shock tube tests, two-dimensional instability tests and forced turbulence simulations in three dimensions. We find subtle differences between the codes in the shock tube tests, and in the statistics of the turbulence simulations. The new Riemann solver increases the computational speed without significant loss of accuracy.

Keywords: Euler equations; Compressible turbulence; Riemann solvers; Godunov methods

1. Introduction

In modern astrophysics the interplay between observations and numerical experiments plays a central role. Typically hydrodynamical flows with high Reynolds numbers and Mach numbers are studied, and they are modelled by the Euler equations

$$\begin{aligned}
&\rho_t + \operatorname{div}(\rho \mathbf{u}) = 0 \\
&(\rho u)_t + \operatorname{div}(\rho u \mathbf{u}) + p_x = \rho f_1 \\
&(\rho v)_t + \operatorname{div}(\rho v \mathbf{u}) + p_y = \rho f_2 \\
&(\rho w)_t + \operatorname{div}(\rho w \mathbf{u}) + p_z = \rho f_3 \\
&E_t + \operatorname{div}((E+p)\mathbf{u}) = \rho \mathbf{f} \cdot \mathbf{u}
\end{aligned} \tag{1.1}$$

Here ρ is the mass density, $\mathbf{u} = (u, v, w)$ is the velocity field, p is the pressure, and E is the energy density $E = \frac{1}{2}\rho\mathbf{u}^2 + \rho e$ with e the specific internal energy. External forces are given in units of acceleration by

[*] Corresponding author.
E-mail address: knut.waagan@cma.uio.no (K. Waagan).

0021-9991/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2007.07.034

$\mathbf{f} = (f_1, f_2, f_3)$. The system is closed by the equation of state that relates p to ρ and e. In this work we consider ideal gases where $\rho e = p/(\gamma - 1)$ for some $\gamma > 1$, and isothermal gases. An isothermal gas has constant temperature T, which implies $p = a^2\rho$ for $a = \frac{RT}{\mu}$, with R the gas constant and μ the mean molecular weight. In a real astrophysical flow additional physical phenomena such as magnetic fields, gravitational forces and electromagnetic radiation may be important, but this paper is only concerned with the hydrodynamics. The specific physical entropy s is defined by the relation

$$de + p\,d\frac{1}{\rho} = T\,ds \tag{1.2}$$

with $T = T(\rho, e) > 0$ the temperature. The second law of thermodynamics implies that

$$(\rho\phi(s))_t + \text{div}(\rho\mathbf{u}\phi(s)) \leqslant 0 \tag{1.3}$$

for any smooth, nonincreasing and convex ϕ. In high Mach number flows this condition is needed to ensure the dissipativity of shocks, since the viscous forces are ignored in (1.1).

To numerically solve (1.1), shock-capturing finite volume schemes are widely used. In astrophysics it is often done with the PPM algorithm described in [4], often with an iterative method to approximate the exact midpoint value of the Riemann fan. This was implemented in the Prometheus code in 1989, using the iterative Riemann solver of [3] with a fixed number of iterations, see [8]. An efficient, parallelised version was then implemented in 2001, see [12,13]. Stochastic forcing for turbulence simulations was added later, see [16,18]. Results produced by the Prometheus code have been presented in many astrophysical publications, for example in [10,12,13,15]. We used this code from 2001 as the basis for this work, and implemented our changes into it.

First, we switched the Riemann solver to an HLLC solver with the signal speeds of Bouchut [1,2]. As far as we know, this is the first implementation of this advancement into an astrophysics code. It is well known that approximate Riemann solvers like HLLC are computationally efficient and easy to implement. In addition, the Riemann solver of Bouchut has two good properties: (a) It automatically ensures that a discrete version of the entropy inequality (1.3) holds. (b) The density and pressure stay positive. Often, finite volume codes need to check for each cell update whether the new data are physically reasonable, but with these two a priori estimates, no checks are necessary apart from underflow treatment. The two estimates are true in the first order case.

When using this Riemann solver in a higher order scheme, these two properties are not automatically inherited. Hence, we introduced a piecewise linear reconstruction, and replaced the characteristic back-tracing with Runge–Kutta time integration. This was done in such a way that positivity is preserved at one half the CFL-number required in the first order case. Such a second order scheme which also satisfies the entropy inequality is however impractical, see [5], but entropy stability for first order schemes has so far seemed to be a good condition in practice. A different notion of stability comes from scalar conservation laws, which have solutions with nonincreasing total variation. This notion is also important in the design of higher order methods for systems. The reconstruction algorithm and the Runge–Kutta integration we use ensure a nonincreasing total variation when applied to scalar equations.

This gave rise to four codes as summarized in Fig. 1. The codes RK-HLLC and PPM-HLLC both run about 20% faster than PPM on the same data with the same resolution. However, the difference between

	Riemann solver:	
Higher order algorithm:	Iterative of Prometheus	HLLC-Bouchut
PPM reconstruction and back-tracing	PPM	PPM-HLLC
Piecewise linear in space, Runge-Kutta in time	RK-exact	RK-HLLC

Fig. 1. The table summarises the four codes we tested. Along the horizontal axis the Riemann solver changes, while vertically the higher order algorithm varies.

the algorithms might be larger, since only the original PPM-code was optimised. The RK-exact code is the slowest, but it was of fundamental interest in this project to compare its accuracy to the RK-HLLC code's.

In the remainder of this introduction we will first sum up the main ideas of the underlying PPM algorithm in the Prometheus code. Then we will describe the new Bouchut–HLLC solver and its theoretical advantages. The second order algorithm is outlined next, and we show how it preserves positivity. In Sections 2–4 the one-, two- and three-dimensional test comparisons are presented. At the end there is a conclusion.

1.1. PPM and the Prometheus code

The basis of the Prometheus code is the one-dimensional PPM-method of [4] with the iterative Riemann solver from [3]. Strang splitting is then used to handle multidimensions. The crucial point of PPM is the so-called characteristic back-tracing. This technique produces a second order approximation to the states at the cell interfaces at the half time step, allowing the use of the midpoint method in time. These approximate states are then used as input to the Riemann solver. Although the overall accuracy is second order, the spatial reconstruction is piecewise parabolic, which is reported to give better resolution than piecewise linear reconstruction. Furthermore, the accuracy at contact discontinuities is improved by an algorithm that detects them, and then steepens the reconstructed density. There is also an algorithm that adds artificial diffusion in order to avoid oscillations behind strong, slow-moving shocks without smearing out the solution much. The reconstruction is required to be monotocity preserving. This means that the order of the scheme may drop locally at extremal points of a reconstructed quantity, which means a primitive quantity in the case of PPM as in [4]. This drop in order is also a feature of the second order reconstruction we use with the Runge–Kutta time integration.

In order to resolve shocks and shock interactions a Riemann problem is solved with the data from the backtracing operation as input. For the Euler equations there is no general explicit formula for the solution of the Riemann problem, so in Prometheus an iterative procedure provides instead approximate values of the fluxes at cell interfaces. For efficiency the number of iterations is limited to a fixed number.

1.2. The HLLC solver of Bouchut and stability

The notion of an approximate Riemann solver goes back to the Roe solver, see [14], which is based on a local linearization of the fluxes at the cell interface. We refer to [21,2] for a modern presentation of the basic ideas. The basic idea is to replace the exact Riemann fans in Godunov's method with something simpler that still gives a numerical flux that is consistent and conservative. In addition entropy consistency and positivity of density must be somehow ensured. For this linearized solvers require additional treatment, a so-called entropy fix.

The simplest approximate Riemann solver is the HLL solver, where the Riemann fan is replaced by a constant state separated by two discontinuities moving with constant speeds C_l and C_r. A sufficient condition for stability is that the exact Riemann solution does not have waves with speed outside the interval $[C_l, C_r]$. The main weaknesses of this approach is that material contact waves are smeared out, and that the signal velocities C_l and C_r have to be guessed. A solution to the first problem was hinted at already in [9], and was carried out in [20], see also [21], with the so called HLLC-solver. The HLLC-solver consists of three discontinuities traveling with speeds C_l, u^* and C_r, where velocity and pressure are held constant across the middle wave, and u^* is this intermediate value of the velocity.

In [2], the HLLC-solver was improved by showing that it results from a relaxation system, which established its entropy stability. Furthermore, sharp explicit formulas for the signal speeds that ensure positivity and entropy stability could be given. We refer to these as Bouchut speeds, and they are given by formula (2.133) and Proposition 2.18 of [2].

1.3. The new second order algorithm

When going to higher order, requiring the reconstruction to be entropy dissipative leads to impractical methods, but there is a way to preserve positivity in a rigorous manner. In the rewritten version of Prometheus RK-HLLC and RK-exact we used the following reconstruction, based on [2]

As a limiter we use

$$\operatorname{minmod}(a_l, a_r) = \begin{cases} 0, & a_l a_r \leqslant 0 \\ \operatorname{sign}(a_l) \min \alpha|a_l|,\ \frac{1}{2}(|a_l| + |a_r|), \alpha|a_r|, & a_l a_r > 0 \end{cases} \tag{1.4}$$

with α set to 1.4. This is applied to produce the discrete differential

$$D\rho_i = \frac{1}{h}\operatorname{minmod}(\rho_{i+1} - \rho_i, \rho_i - \rho_{i-1}) \tag{1.5}$$

and Du_i, $D(u_\perp)_i$, $D(\rho e)_i$ similarly. The positivity of the reconstructed density is guaranteed since $\rho_i - \frac{h}{2}|D\rho_i| > \min_k(\rho_k)$. Conservation of momentum dictates that we take the reconstruction slope

$$D(\rho u)_i = \rho_i Du_i + uD\rho_i - \frac{h^2}{4} D\rho_i Du_i \tag{1.6}$$

and similarly for $D(\rho u_\perp)_i$. Energy is conserved by replacing e_i with

$$\tilde{e}_i = e_i - \frac{h^2}{8} 1 - \frac{h^2}{4\rho_i^2} D\rho_i^2 Du_i^2 + (Du_\perp)_i^2 \tag{1.7}$$

when computing the reconstructed internal energy. The extra terms cancel out the conservation errors in kinetic energy caused by the linear reconstruction. Hence positivity means that $\rho_i \tilde{e}_i - \frac{h}{2}|D(\rho e)_i| > 0$, or in other words

$$\frac{h^2}{8} Du_i^2 + (Du_\perp)_i^2 < \frac{1}{1 - \frac{h^2}{4\rho_i^2} D\rho_i^2} \left(\rho_i e_i - \frac{h}{2}|D(\rho e)_i|\right) \hat{=} \Lambda_i^2 \tag{1.8}$$

To ensure (1.8) in a consistent way we first restrict $|D(u_\perp)_i|$ to less than or equal to Λ_i, and then set Du_i^2 less than or equal to $\Lambda_i^2 - |D(u_n)_i|^2$. Note that in practice we multiply Λ_i with a number slightly less than one to ensure that the inequality (1.8) is strict.

We did not apply any special treatment of material contact waves in this code version, and no artificial diffusion was added at shocks.

The numerical time integration is a second order Runge–Kutta method. That is, one does two full time steps, and then averages the resulting cell average with the initial one. This procedure preserves positivity, and is total variation diminishing. Multidimensionality is taken care of by Strang splitting just as in the PPM-codes.

2. One space dimension: shock tube tests of Toro

A basic setup for testing these methods are one-dimensional Riemann problems, or shock tube tests. In [21], five very useful such test problems are given and subjected to several different Riemann solvers. The problems are carefully devised to exhibit phenomena known to be hard to reproduce numerically.

As reference solutions we simulated all tests with 10^5 grid cells using the original PPM-code. In some cases this was not locally converged due to spurious oscillations, etc., and we point out these anomalies when they occur. In all the runs the CFL-number was 0.4, and we considered $x \in (0, 1)$ with a resolution of 100 grid cells.

2.1. Test 1

The first test is not the most severe, but it contains a transsonic rarefaction, which nonentropic schemes have trouble with. The initial data are

$$(\rho, u, p) = \begin{cases} (1, 0.75, 1), & x < 0.5 \\ (0.125, 0, 0.1), & x > 0.5 \end{cases} \tag{2.1}$$

All schemes handle the transsonic rarefaction without any signs of a nonentropic glitch, but there are differences in the resolution at the rear end of the rarefaction with the PPM doing the best job. However, PPM gives large oscillations behind the contact discontinuity compared to the other codes. With the RK-codes there is little difference between the Riemann solvers. We note the undershoot in front of the contact, and the less sharp resolution of the contact compared to PPM (see Fig. 2). These observations also hold true at increasing resolution, as illustrated by the L^1-errors in mass density in Fig. 3.

2.2. Test 2

Test 2 has two rarefractions going apart creating a low density region. The initial data are

$$(\rho, u, p) = \begin{cases} (1, -2, 0.4), x < 0.5 \\ (1, 2, 0.4), x > 0.5 \end{cases} \tag{2.2}$$

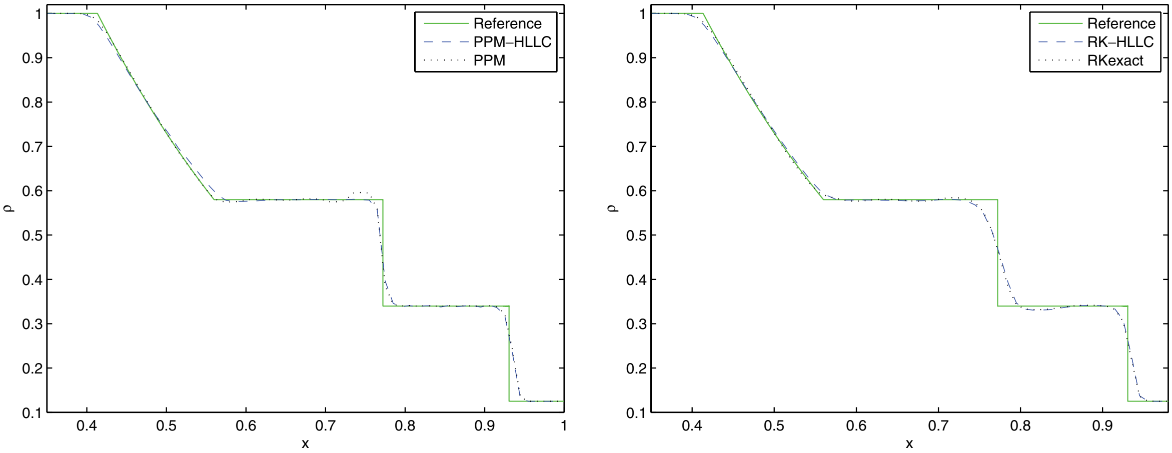


Fig. 2. Results for Toro test 1.

Resolution:	PPM	PPM-HLLC	RK-HLLC	RK-exact
$h = 0.01$	3.75	3.11	5.07	5.60
$h = 0.005$	1.48	1.61	2.63	2.92
$h = 0.0025$	0.88	0.93	1.73	1.85
$h = 0.00125$	0.42	0.39	0.95	0.98

Fig. 3. The table shows the L^1-error in the computed mass density of Toro's test 1 for different codes and resolutions. The errors are given in units of 10^{-3}.

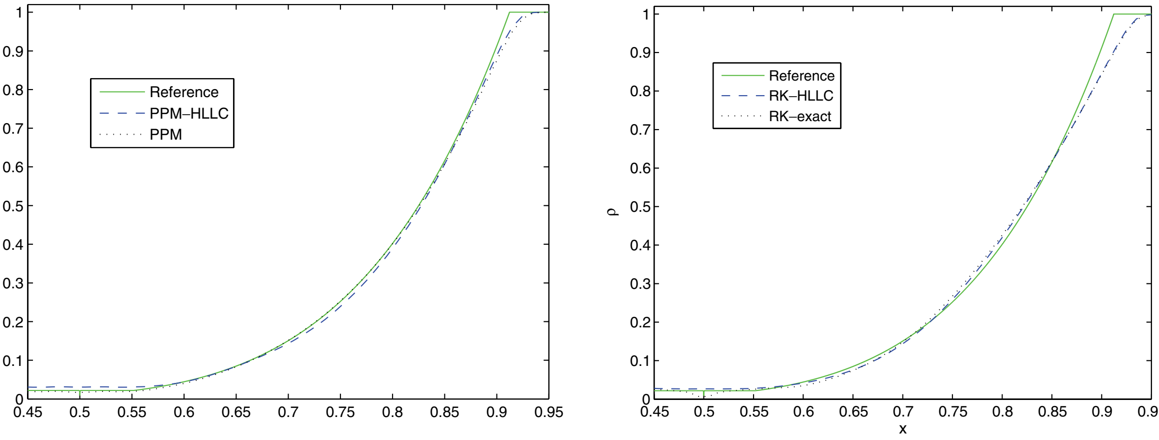


Fig. 4. Results for Toro test 2. The results are symmetric around $x = 0.5$.

The solver should be able to handle this without giving negative density or pressure. In particular, linearized solvers have trouble with such cases. In the density plots, Fig. 4, we note a bump in the density at $x = 0.5$ with the RK-exact code. We see similar tendencies for the PPM simulation, and in the PPM reference solution there is a deep narrow bump.

For the RK-HLLC-code positivity was automatically maintained, and it is interesting that we get a better approximation of the density value in the middle compared to PPM-HLLC, and also of the velocity, Fig. 5. The front of the rarefaction is however best resolved by the PPM-codes.

Notice in Fig. 5 that both PPM and RK-exact (which have the same Riemann solver) has oscillations in the velocity near $x = 0.5$. The RK-exact code especially had problems with this test, and positivity had to be artificially imposed for CFL-numbers larger than around 0.05. Theoretically, a CFL-number less than 0.25 should ensure positivity with an exact Riemann solver, so this has to do with the iterative procedure in the Riemann solver not automatically ensuring the positivity property. With the iterative solver as part of PPM however, this seemed not to cause serious problems.

2.3. Test 3

Test 3 is a high Mach number shock tube with initial data

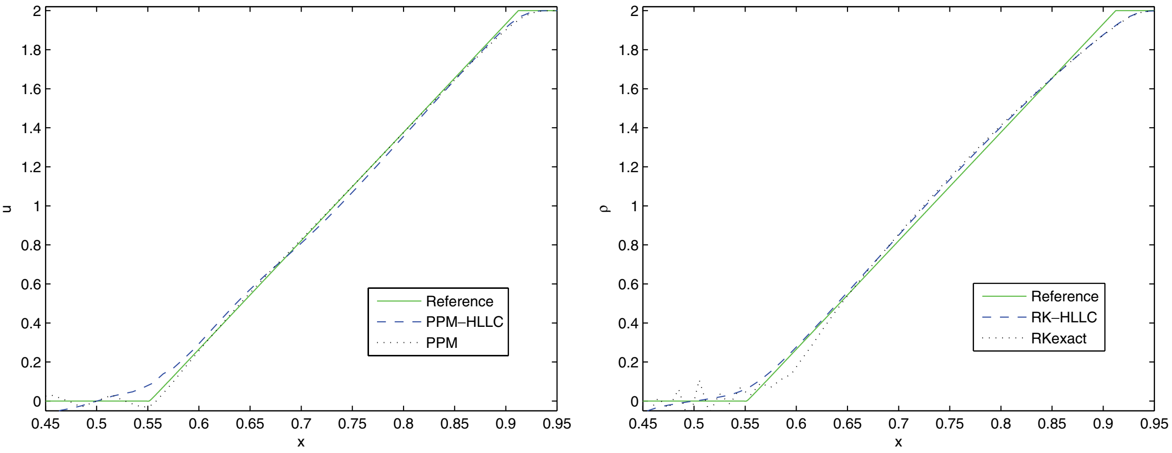


Fig. 5. Results for Toro test 2. The results are symmetric around $x = 0.5$.

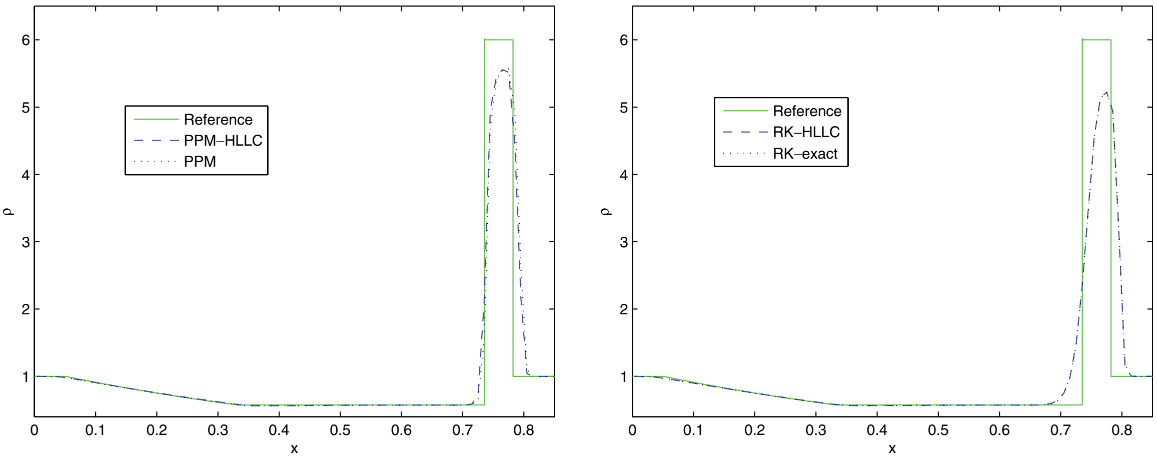


Fig. 6. Results for Toro test 3.

$$(\rho, u, p) = \begin{cases} (1, 0, 1000), & x < 0.5 \\ (1, 0, 0.01), & x > 0.5 \end{cases} \tag{2.3}$$

We found little difference between the codes here apart from the expected slightly sharper resolution of the PPM-codes, which was most prominent on the contact wave (see Fig. 6). All codes produced spurious effects behind the rarefaction, as seen in the overshoot in the velocity plots, Fig. 7.

2.4. Test 4

The solution of test 4 has a near stationary shock, that is, the shock speed is small compared to the characteristic speeds, hence if the numerical diffusion applied to this shock is high, it will be particularly pronounced. The initial data are

$$(\rho, u, p) = \begin{cases} (5.99924, 19.5975, 460.894), & x < 0.5 \\ (5.99242, -6.19633, 46.0950), & x > 0.5 \end{cases} \tag{2.4}$$

Also, oscillations behind such shocks is a well known phenomenon, and in the original PPM paper, a so called ‘flattening’ algorithm was introduced which essentially adds diffusion. This algorithm was used in all the PPM simulations here, but for this test we also tried to switch it off, resulting in oscillations in the density of mag-

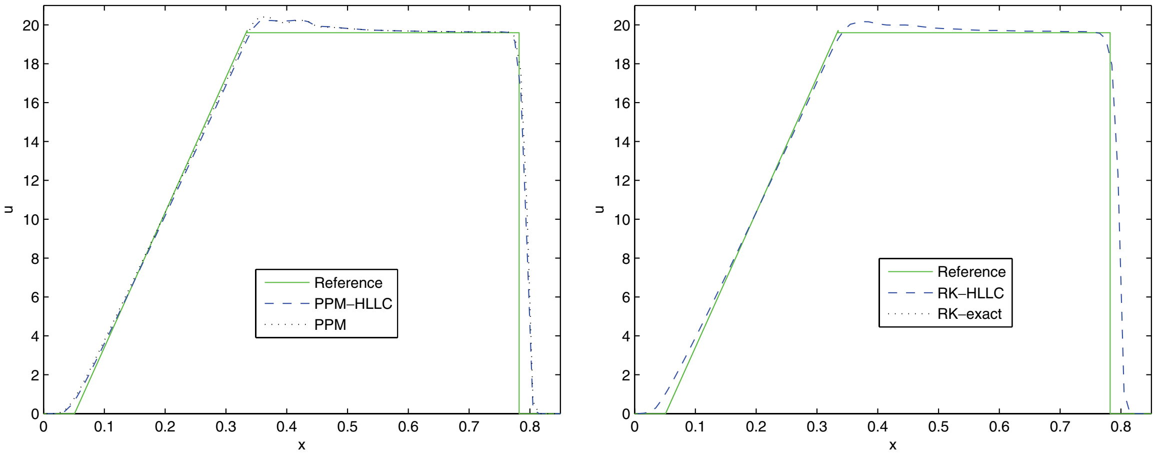


Fig. 7. Results for Toro test 3.

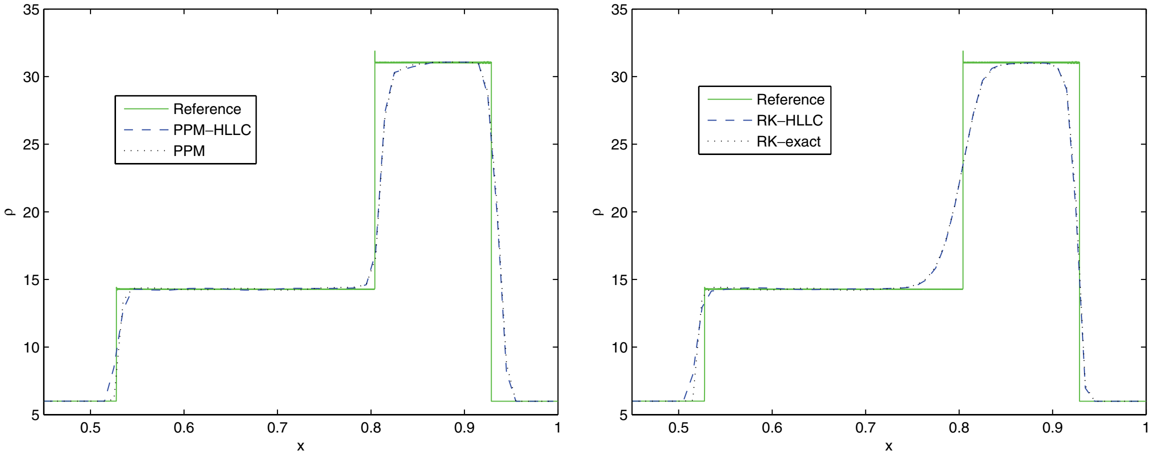


Fig. 8. Results for Toro test 4.

nitude around 5% of the postshock density both for PPM and PPM-HLLC. The RK-codes only show small oscillation here, and no special treatment was necessary. In the reference PPM solution, there are pronounced oscillations both after the near stationary shock as well as between the contact and the right moving shock. Both HLLC-codes smear the near stationary shock out with 1–2 grid cells more in front of the shock, as seen Fig. 8 (we show the results from PPM with flattening). The difference of 1–2 grid cells was maintained when refining to 200 and 400 grid cells. It is probably caused by the signal speeds of HLLC–Bouchut slightly overestimating the shock speeds. Otherwise we only note the lower resolution of the contact wave with the RK-codes compared to HLLC.

2.5. Test 5

Test 5 is like test 3 with a background velocity resulting in a near stationary contact discontinuity. The initial data are

$$(\rho, u, p) = \begin{cases} (1, -19.59745, 1000), & x < 0.8 \\ (1, -19.59745, 0.01), & x > 0.8 \end{cases} \tag{2.5}$$

All codes handle this feature reasonably well, which was expected since both Riemann solvers exactly resolve contact waves, see Fig. 9. Note that in this case the RK-codes have comparable resolution of the discontinu-

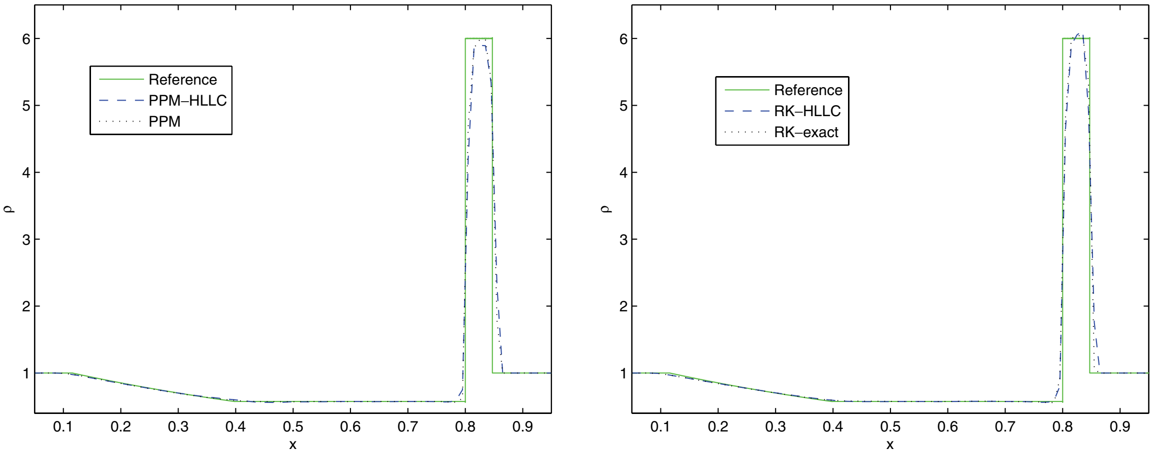


Fig. 9. Results for Toro test 5.

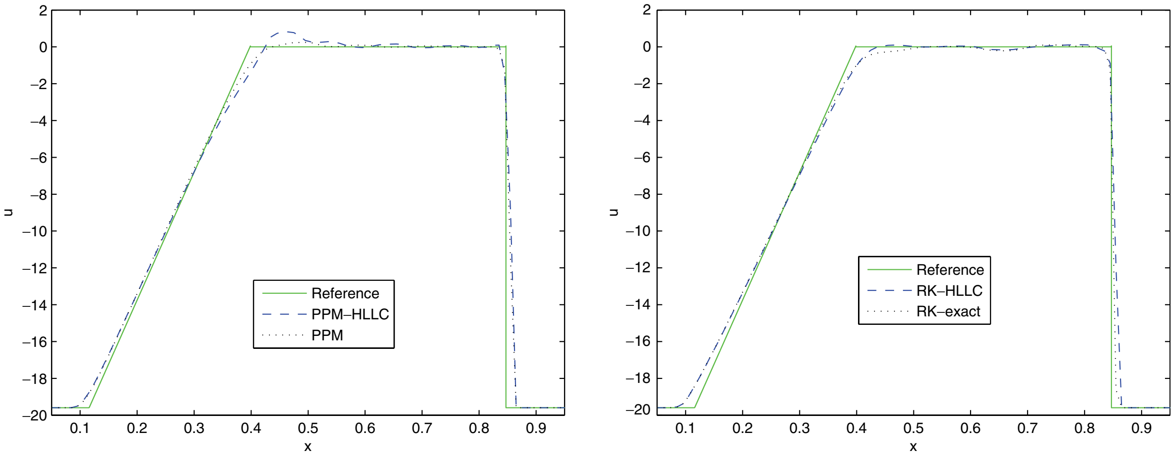


Fig. 10. Results for Toro test 5.

ities to the PPM-codes, but they overshoot the right intermediate density slightly. The oscillations occuring behind the rarefaction in the velocity, see Fig. 10, is not reported with the first order schemes tested by Toro, so it must have something to do with the higher order algorithms. It is especially pronounced with PPM-HLLC, but visible in all simulations.

3. Two space dimensions: mixing layers

We now look at transitions from laminar into unstable flows in two dimensions. By the nature of the underlying unstable flows, although we observe differences in the output, we are not really able to infer much about the quality of the respective codes. However, we observed that the instabilities needed to be highly developed before any differences could be seen between the Riemann solvers. In other words, at the onset of instability the Riemann solvers seemed to give the same results. We recall from the one-dimensional tests that changing the Riemann solver had only a small effect on the numerical smearing. This indicates that the 'numerical viscosity' varies little between the different schemes, and the sensitivity of our instabilities to numerical viscosity seems to be relatively small at their onset.

3.1. Kelvin–Helmholtz instability

Two layers of fluid moving with different parallel velocities are always unstable in the absence of viscosity and external forces. This is referred to as a Kelvin–Helmholtz instability, and seems to be an important source of turbulence in many applications. We consider a grid-aligned jump in velocity here, and make a small periodic perturbation. The initial data are $\rho = 1$, $\gamma = 1.4$, and we let p vary to allow different Mach numbers. The velocity is in the y-direction with $v = 0.5$ for $x < 0.5$, and $v = -0.5$ for $x > 0.5$, however, we moved the velocity jump one grid cell to the left to break the symmetry. We perturb v with $2\pi e^{2\pi x}\cos(2\pi y)/100$ for $x < 0.5$ and $-2\pi e^{-2\pi x}\cos(2\pi y)/100$ for $x > 0.5$. This means we can compute $y \in (0, 1)$ with periodic boundary conditions, and in the x-direction we consider $x \in (0.1)$ with reflecting boundary conditions. The CFL-number was 0.8 in all simulations.

First we take $p = 1/\gamma$, which means that the relative velocity between the layers equals the sound speed. We consider the time history of the average of $\frac{1}{2}\rho u^2$, as this quantity is often used as a measure of the growth of the instability. Fig. 11 shows $\log\overline{\frac{1}{2}\rho u^2}$ as a function of time for the different codes. We used three different resolutions, 100^2, 200^2 and 400^2 points, and we see here that the instability growth rate increases with resolution. This is plausible, since by linear instability theory, the growth rate is inverse proportional to the perturbation

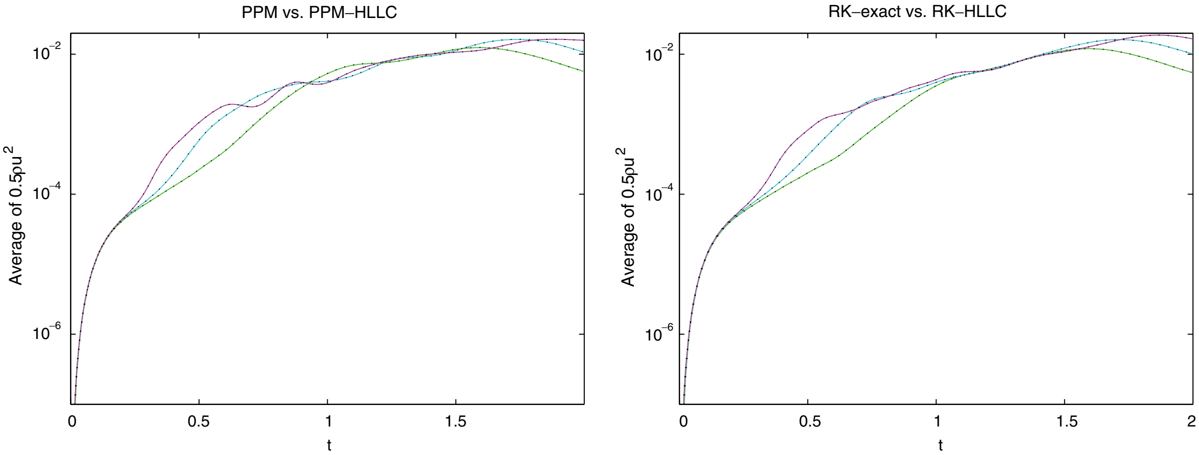


Fig. 11. Growth of transversal kinetic energy component. The codes with exact solver are represented with dotted lines, and the HLLC version with solid lines. We show data from runs with 100^2 200^2 and 400^2 grid cell, the curve steepness increasing with resolution.

wavelength. The Riemann solver, however, seems to have no influence at all. We measured the CPU-times for the different codes for this simulation with 200^2 grid points until time $t = 2.0$. We got the following results:

Code	PPM	PPM-HLLC	RK-HLLC	RK-exact
CPU-time	1.00	0.81	0.79	1.19

These are the averages of two runs with each code. The numbers are normalised to the result from PPM.

Fig. 12 shows the time evolution of the velocity field with PPM-HLLC, illustrated by streamlines at times 0.25, 0.50 and 1.0. The streamlines were produced with the intrinsic Matlab routine ‘streamslice’. Note that the density of streamlines plotted does not accurately reflect the numerical resolution, but were chosen to give a clear representation of the observable topological flow features. The plots from the original PPM looks very similar. Fig. 13 shows the same but this time with the RK-HLLC code. Also with these codes we see no significant differences between the two Riemann solvers.

Differences between the schemes only become apparent at later times. Eventually all vortices are swallowed by the domain-centered vortex, and we see some differences in at which time t_1 this happens. We used the streamline plots to find approximately when this change in flow topology occurs. For example in Fig. 14, we plotted the flow at time $t = 20$, since the PPM-simulation then still had two distinct vortices, while with PPM-HLLC we could only see one. In Figs. 15–17 we do the same with different resolutions and codes. Since the resolution and the underlying code also influenced the time t_1, we chose different plotting times in Figs. 14–17. In all four cases considered it is clear that the Riemann solver induces some difference in t_1.

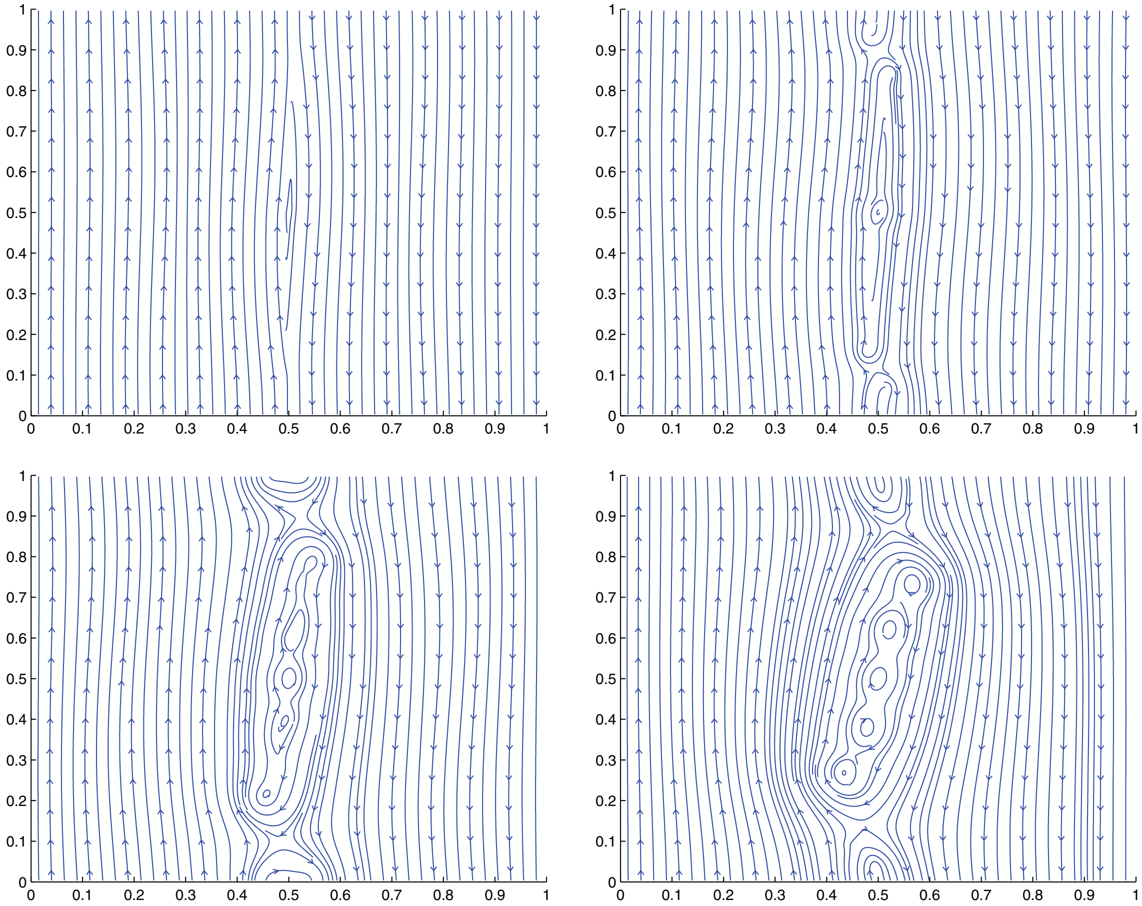


Fig. 12. The development of the Kelvin–Helmholtz instability is illustrated by the streamlines at times 0.25, 0.50 and 1.0 from PPM-HLLC with 200^2 resolution. (*In the plots x is on the horizontal axis, and y on the vertical axis.)

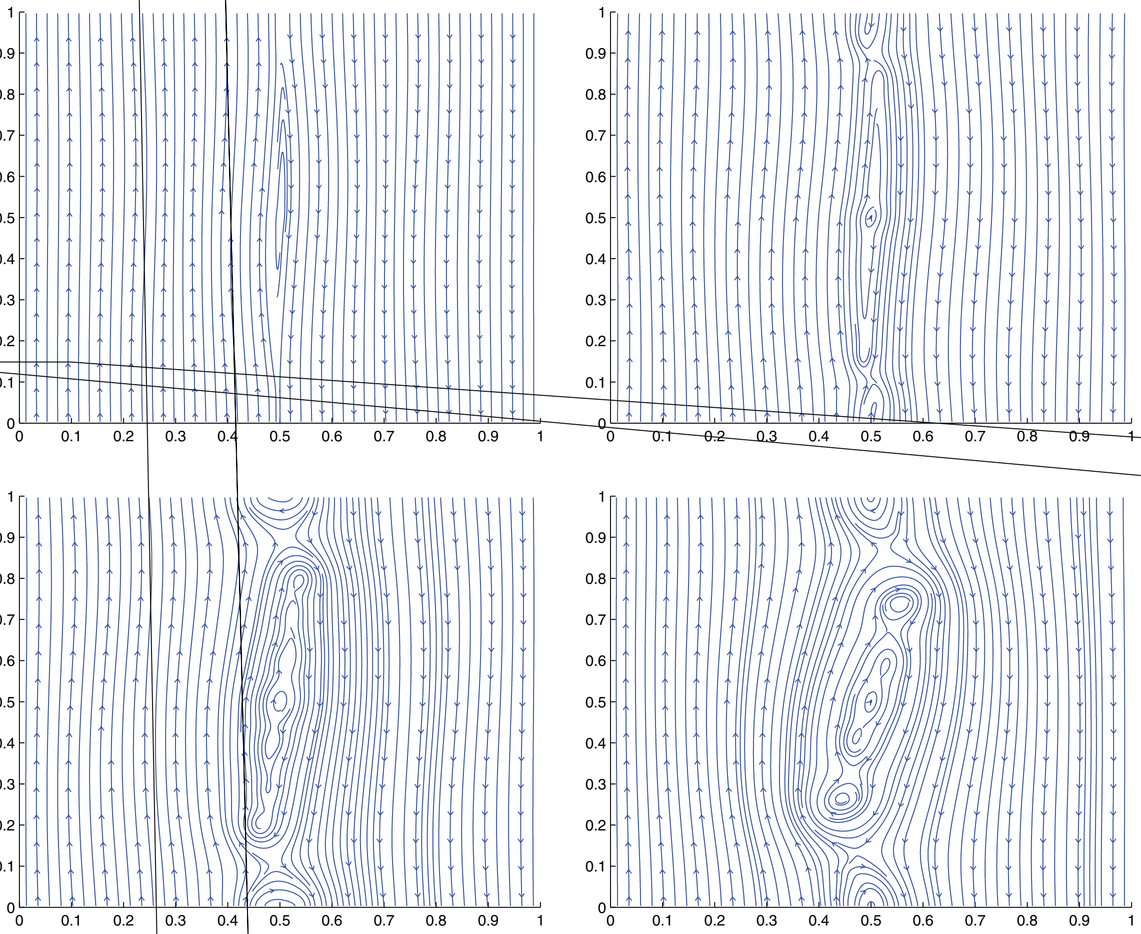
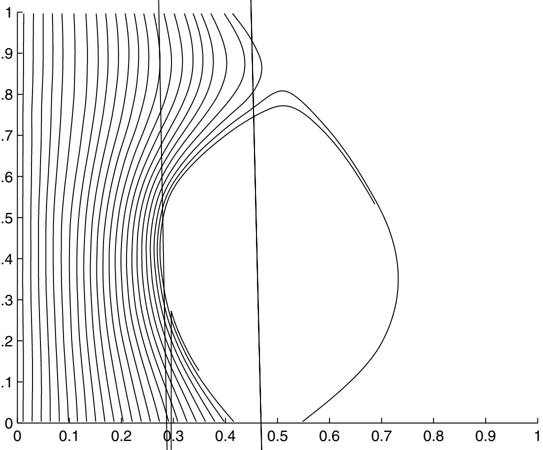


Fig. 13. The same as Fig. 12 with RK-HLLC. (*In the plots x is on the horizontal axis, and y on the vertical axis.)



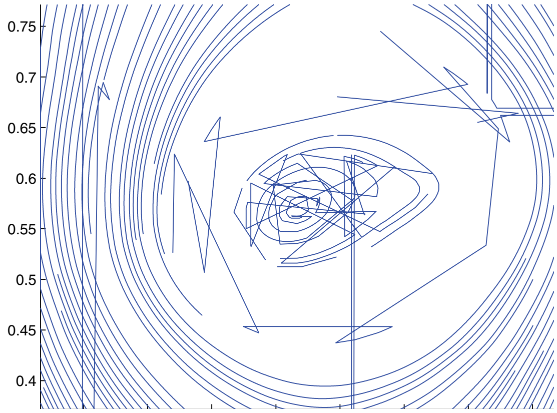


Fig. 17. Streamlines from simulations with 400^2 cells at time $t = 16$. RK-HLLC is shown on the left, and RK-exact right. (*In the plots x is on the horizontal axis, and y on the vertical axis.)

We also considered $p = \frac{1}{100\gamma}$, giving a relative Mach number of 10. Fig. 18 shows the growth of the average transversal kinetic energy component. The effect of the Riemann solver is indiscernible. Resolution had less influence here than with relative Mach number 1, so we only show the data from the 200^2-simulation. At Mach numbers this high, the Kelvin–Helmholtz modes are linearly stable, and instead of Kelvin–Helmholtz rolls, kink modes develop, see for example [11]. For the Mach 10 case differences are also very subtle. In Fig. 19, showing filled density contours at time $t = 10$, one can see slightly more small scale structure with PPM than PPM-HLLC by careful inspection of the plots. The similar plots from the RK-codes in Fig. 20 clearly shows more smeared out structures than the PPM-codes. There is no noticeable difference between RK-HLLC and RK-exact, which means that any effect of changing the Riemann solver is much less prominent than the smearing due the RK-algorithm. The superimposed density contours in Fig. 21 also illustrate the increased numerical diffusion of the RK-algorithm, and that this suppresses the effect of changing the Riemann solver.

3.2. Richtmeyer–Meshkov instability

The Richtmeyer–Meshkov instability occurs when a planar shock hits a parallel, slightly perturbed density jump. We used the following setup to simulate this. The domain was $(x, y) \in (0, 16) \times (0, 1)$ with periodic

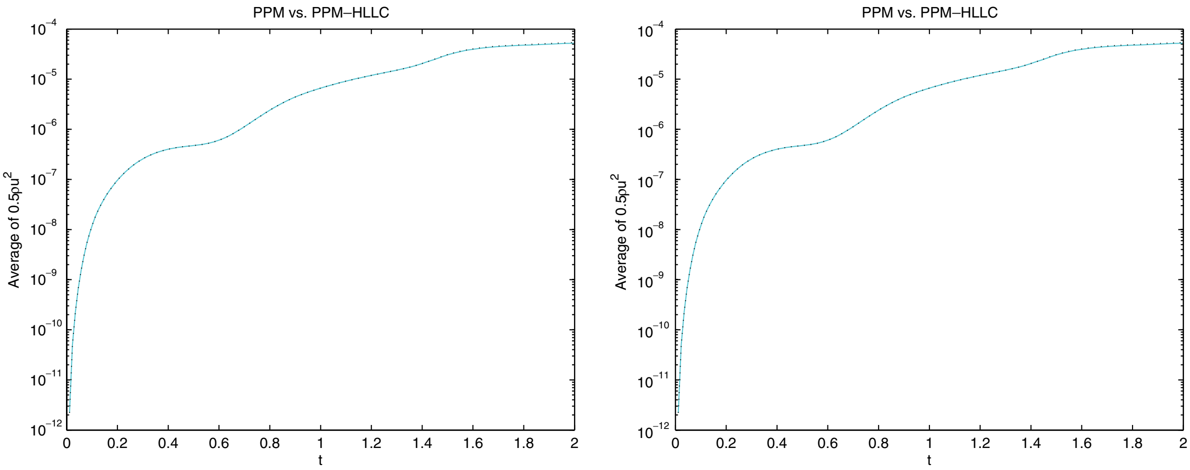
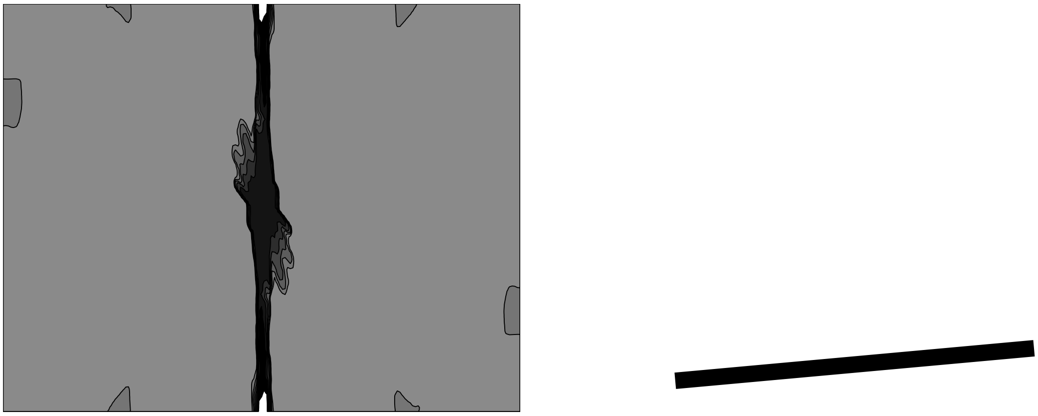
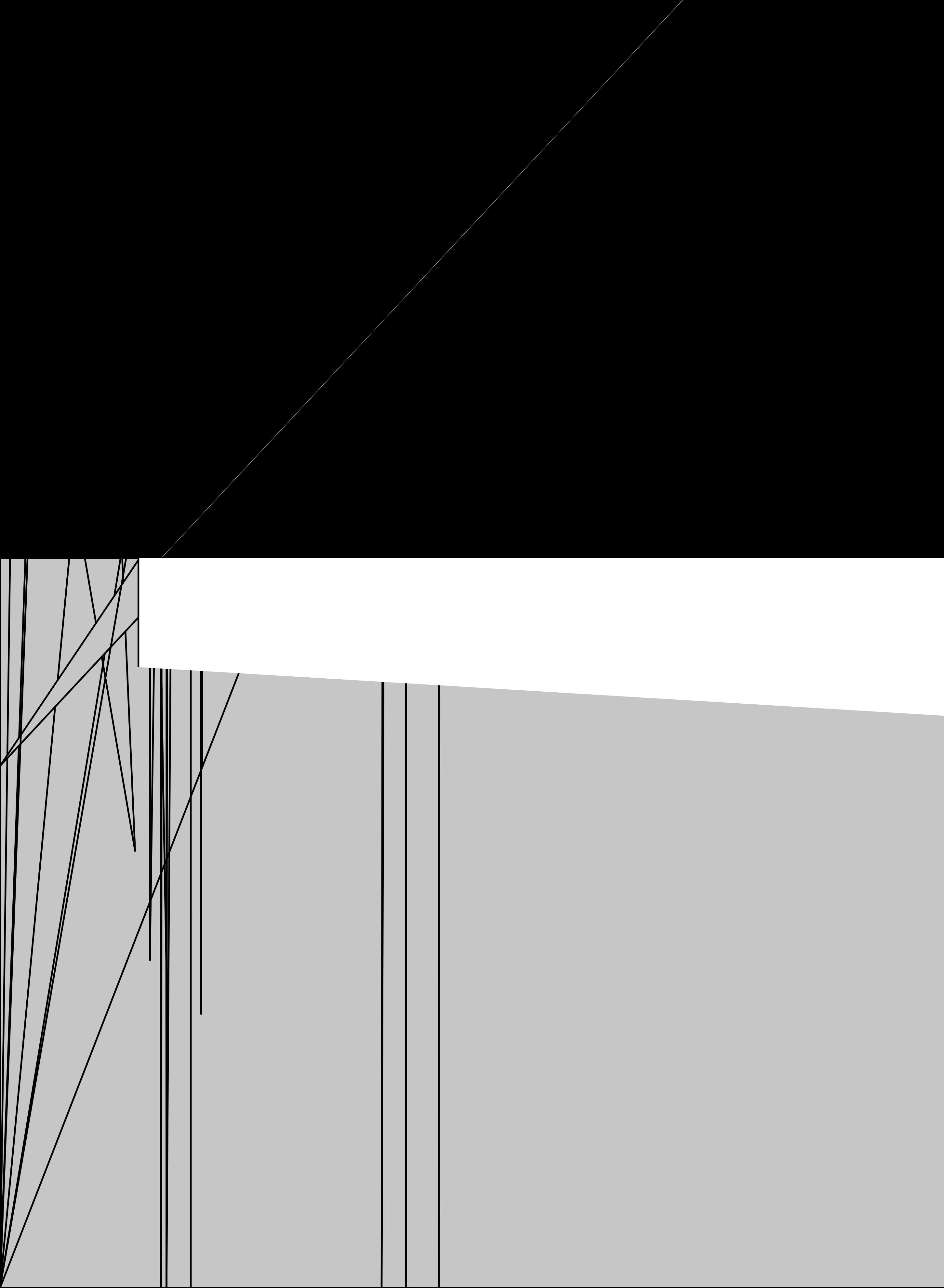


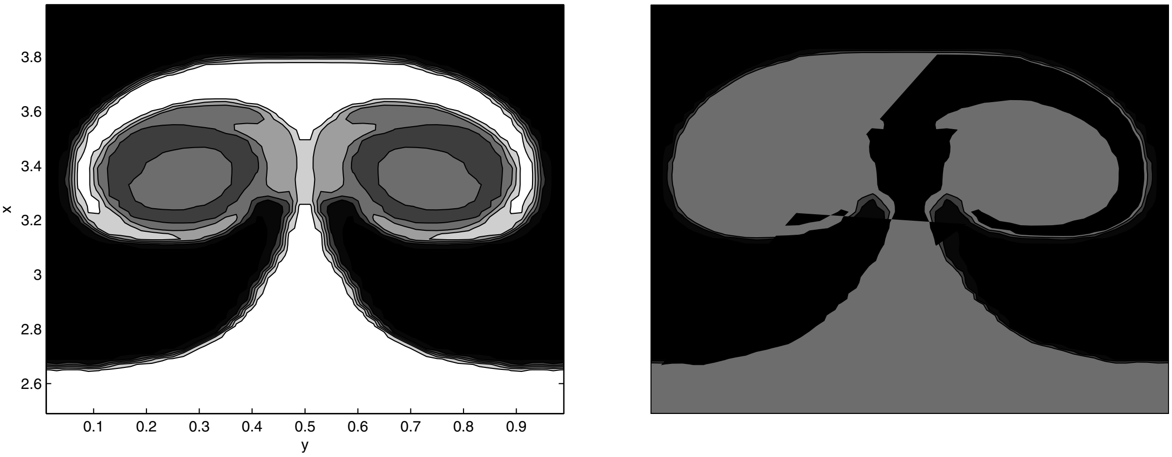
Fig. 18. Growth of transversal kinetic energy component with relative Mach number 10. The codes with exact solver are represented with dotted lines, and the HLLC version with solid coloured lines. They look the same.





boundary conditions in y, and Neumann boundary conditions in x. We set up a shock tube problem in the x-direction at $x = 1.6$ with density and pressure as in the first Toro test and constant velocity $u = -1$ in the x-direction. We considered adiabatic gas $\gamma = 1.4$ so as to coincide with the three-dimensional runs rather than the shock tube tests. At the line $x = f(y) = 3.2 + 0.2\cos(2\pi y)$, the density fell by a factor of 2. When the shock goes through the initial density jump, the boundary $f(y)$ evolves into a mushroom-like structure. In Fig. 22, we see a slice in x-direction of the density profile at time $t = 1.0$. The shock has just hit the density jump, and we see a weaker shock going through, and a reflected wave moving back towards the contact from the shock tube problem. This last wave might cause some minor reflected waves, but otherwise the instability is not influenced by other features. The boundary at $x = 0$ is transparent to the supersonic rarefaction, and we stop before the shock reflection at $x = 16$ affects the instability. The CFL-number was 0.8. Again it is hard to observe differences, see Fig. 23, but it seems the original PPM resolves the 'extremities' of the high density region a bit sharper, at least they extend more.

We also illustrate the growth of the instability here by the time history of the transversal component of kinetic energy in Fig. 24. With RK-HLLC the density jump is more smeared out before the shock hits it, which explains the less steep slope.



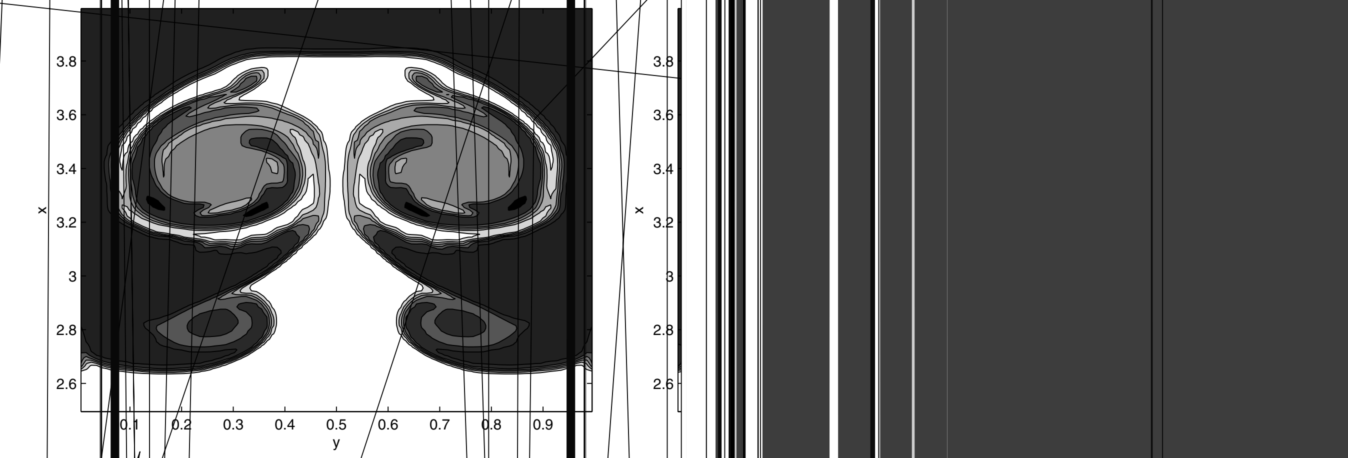


Fig. 25. The same as in Fig. 23, but with the contact wave steepening t

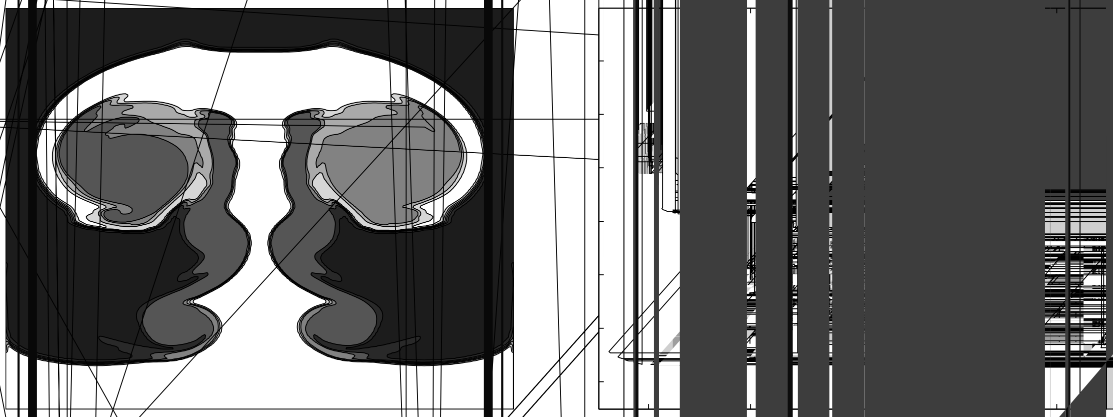


Fig. 26. The same as in Fig. 23, but with the contact wave steepening turned off = Δ y 05.

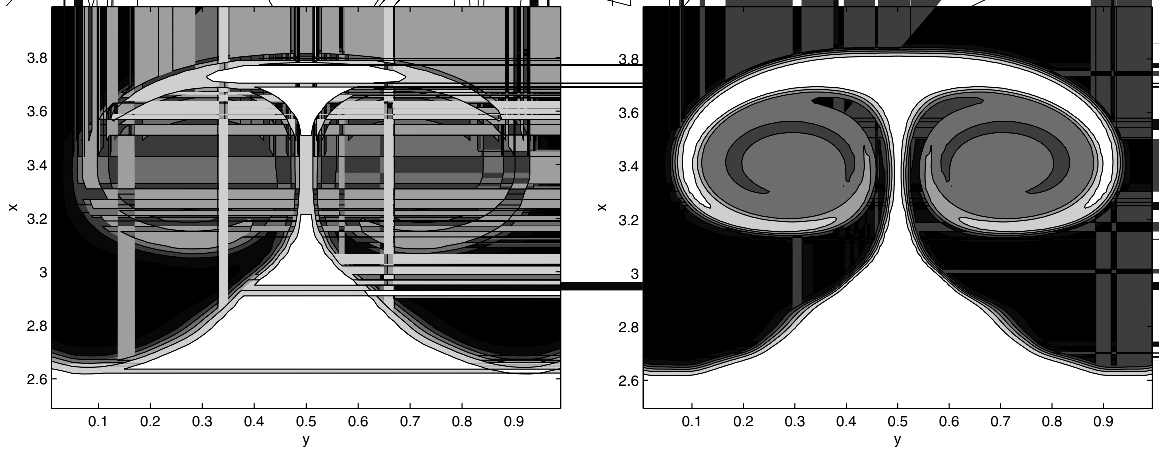


Fig. 27. The same as in Fig. 23 with RK-HLLC. On the left with resolution as in Fig. 23 and on the right with resolution as in Fig. 25.

It is known that the contact wave steepening of PPM may artificially induce instabilities in certain cases. For the PPM-codes we observed small scale structures that we believe to be numerical noise when repeating the simulation on finer grids. However, by switching off the steepening, we got reasonable results. In Figs. 25 and 26, we compare versions of PPM and PPM-HLLC without steepening. With the highest resolution the density profiles differ strongly, but there is no way to tell which code is better.

The RK-HLLC-code produces a more smeared out structure, see Fig. 27.

4. Forced isotropic turbulence

In many real life flows turbulence is an important feature. Since we do not know how to infer from simpler test cases how a numerical method will treat turbulence, we now consider simulations of actual three-dimensional turbulence. Because of the three-dimensional nature of turbulence, to get useful results one needs powerful computational resources, and we were able to perform some parallel simulations on the Hitachi SR8000 at the Leibniz Computing Centre in Munich. The simulations were part of a larger study on parameters in supersonic turbulence, see [17]. We considered the same type of forced isotropic turbulence experiments described in [18]. The resolution here was 256^3 equilateral grid cells, and the boundary conditions periodic as in [18]. We refer to [16,18] for details of the experiments and the analysis tools. In addition to [18], compressible turbulence simulations with PPM have been investigated by Sytine et al. [19].

The tests consisted of a constant, zero velocity initial state continuously subjected to a stochastically varying force field $\mathbf{f}$. The forcing was given by evolving its Fourier transform by a so called Ornstein–Uhlenbeck process, which is a statistically stationary stochastic process, with parameters such that the resulting force was statistically isotropic. Only the larger wavelengths were given a nonzero contribution. Note that the Fourier transform of a periodic function can be understood as a generalized function given by the coefficients in its Fourier series. By varying the magnitude of the forcing, the characteristic velocity of the flow was varied correspondingly. The forcing also had a free parameter ζ corresponding to a projection operator regulating the solenoidality of the force field. For $\zeta = 1$ the force field is divergence free, and for lower values we have progressively stronger compressive force components. We will not study the influence of this parameter here, just note that all flows considered were highly compressible. How a gas responds to this injection of energy depends a lot on the equation of state, as we will show.

Since these flows are highly sensitive to perturbations, it makes no sense to compare the actual solutions. Instead we will compare statistical properties of the simulated flows, since the statistical approach has been relatively successful in quantitatively describing turbulence, see for example [7]. Note also that each simulation represented a different realisation of the stochastic forcing process. One way to extract statistical information is to make a histogram of the different values assumed by a scalar quantity at a fixed time. We can call this to make a probability distribution function (PDF). We will consider PDFs for ρ and the absolute value of the vorticity ω.

As an indicator of numerical dissipation we will look at the energy spectra, that is, we will look at the energy content in each Fourier mode of the velocity field. Parseval's theorem says that the total specific kinetic energy equals the integral over the square of the Fourier transformed velocity field $\hat{\mathbf{u}}(\mathbf{k}, t)$,

$$\int |\mathbf{u}(\mathbf{x}, t)|^2 \,\mathrm{d}\mathbf{x} = \sum_{\mathbf{k}} \hat{\mathbf{u}}(\mathbf{k}, t) \cdot \hat{\mathbf{u}}(\mathbf{k}, t)^* \tag{4.1}$$

where $\cdot^*$ denotes complex conjugation. In other words, it is given by integrating over the energy spectrum function $E(k, t)$, which is defined as the sum of the squares of the Fourier coefficients corresponding to each mode where the three-dimensional wave number vector $\mathbf{k}$ has absolute value k,

$$E(k, t) = \sum_{|\mathbf{k}|=k} \frac{1}{2} \hat{\mathbf{u}}(\mathbf{k}, t) \cdot \hat{\mathbf{u}}(\mathbf{k}, t)^* \tag{4.2}$$

times a scaling factor. We refer to [16,18] for how this was done numerically.

It is intuitively clear that if the solution has a lot of small scale structure, it indicates low numerical diffusion, although spurious oscillations could also play a role. The energy spectrum function gives a way to quantify this idea for these highly complex flows, but it is also connected to deeper ideas about turbulence, in particular Kolmogorov's theory, see for example [7].

Typically a plot of $k \mapsto E(k,t)$ will show three different ranges. For the lowest wave numbers the stochastic injection of mechanical energy dominates. Then comes what is called the inertial range, where Kolmogorov's theory predicts that $E(k,t)$ drops off as $k^{-\frac{5}{3}}$, due to the transfer of energy from vortices of higher to lower length scales. This 'Kolmogorov cascade' has been observed for low enough Mach numbers both in experiments and numerical simulations. For the highest wave numbers numerical dissipation becomes dominant, and $E(k,t)$ drops off steeply. Between the inertial range and the dissipation range, one tends to observe a flattening of $E(k,t)$ in numerical simulations. This is called the bottleneck effect and it is still debated whether it has physical significance, or whether it is a purely numerical effect, see [6,18] and the references in these. With the resolution here of 256^3 cells, the injection range goes straight into a bottleneck range. Since the Kolmogorov theory is derived for incompressible flow, we also define the transversal energy spectrum $E_{\mathrm{tr}}(k,t)$ which only consist of the part of $\hat{\mathbf{u}}(\mathbf{k},t)$ orthogonal to $\mathbf{k}$, so that we only take into account the divergence free part of the velocity.

Some dimensional quantities need to be defined first, but we choose not to go into detail about the physical scales as they are not relevant to the code comparisons. Asymptotically the RMS (root mean squared) amplitude of the force f would approach $(1-2\zeta+3\zeta^2)F_0$ for some prescribed value F_0. We use this to define the characteristic velocity V by $V=(F_0L)^{1/2}$, where L is half the length l of the sides of the periodic box. Hence V is close to the RMS velocity in the fully developed flow. With characteristic Mach number 'Ma' we refer to the ratio of V to the initial sound speed. The simulations were run for five integral time scales $T=\frac{L}{V}$. The forcing is strongest at wave numbers $\mathbf{k}$ such that $|\mathbf{k}|=k_0=\frac{2\pi}{L}$, and zero for $|\mathbf{k}|\geqslant 2k_0$. With α we refer to the integer $\frac{k_0 l}{2\pi}=2$, and the initial density is denoted by ρ

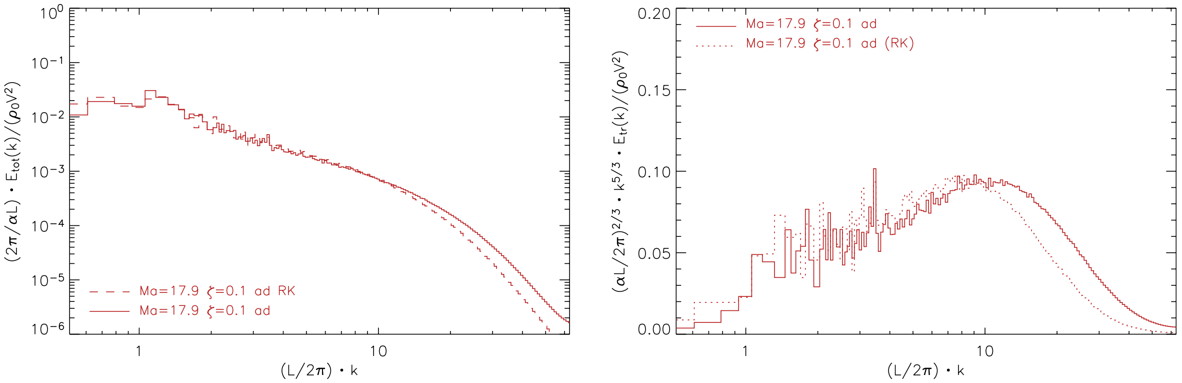


Fig. 29. Energy spectrum at the final time $t = 5T$ for the adiabatic runs (left), on the right the compensated transversal spectrum. The curves are labelled as in Fig. 28.

the RK-HLLC code. Also note the clear bottleneck effect, which is best seen in the plot of the 'compensated' transversal energy spectrum function $\Psi(k,t)$ proportional to $E_{\mathrm{tr}}(k,t)k^{\frac{5}{3}}$ in Fig. 29. The Kolmogorov theory predicts that Ψ should be constant in the inertial range, and then drop off in the dissipation range.

If we look at other times, however, things become more complicated: In Fig. 30, we see the evolution of the energy spectra.

The fact that in Fig. 28, the curves differ up to time $t = 2T$, we attribute to the different realisations of the stochastic forcing. Hence one should compare statistics from the different codes only for times $t > 2T$. The

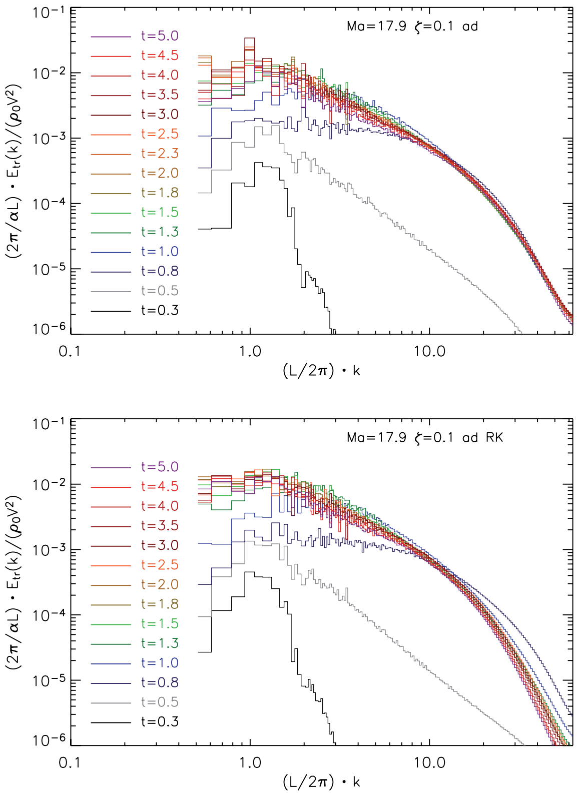


Fig. 30. Time history of transversal energy spectra. Data from PPM are on top, and from RK-HLLC underneath. Times are given in units of integral time scale T.

RK-HLLC-code gave decreasing energy spectra in time (after $t = T$), while the energy spectra from PPM have no particular time dependency (after $t = T$). We interpret this as an increase in dissipativity of RK-HLLC in time, and we associate it with the decrease in Mach number. The observed independence of Mach number for PPM is related to the known fact that this scheme retains, and even improves, its accuracy as the advective Courant number decreases. This is caused by the high order upwind advection used in PPM. The typical advective Courant number in a cell will decrease with the RMS Mach number in these turbulence tests, hence PPM should perform well at the lower Mach numbers.

Fig. 31 shows mass density PDF at different times. There are clear differences between the simulations in both the high and low density regions, but this seems to be due to fluctuations inherent in the stochastic process behind the forcing, as there is no clear trend. The vorticity PDF's in Fig. 32 shows the same tendency as the energy spectra. From time $t = 2T$, the tails in the vorticity PDF's from PPM are clearly longer than those of RK-HLLC, meaning that the flow contains more small scale vortices.

4.2. Isothermal gas

We also performed simulations with isothermal gas. Here the Mach number stays near constant after the initial growth phase, as seen in Figs. 33 and 34. For this reason, we only analyse the data from the final time. We found no significant difference between RK-HLLC and PPM, and there was even less difference between the two PPM-codes. We show PDFs and energy spectra from simulations with Mach numbers 2.1 and 21.1 in Figs. 35–37. Again bottleneck effects are seen in all simulations, see Fig. 38.

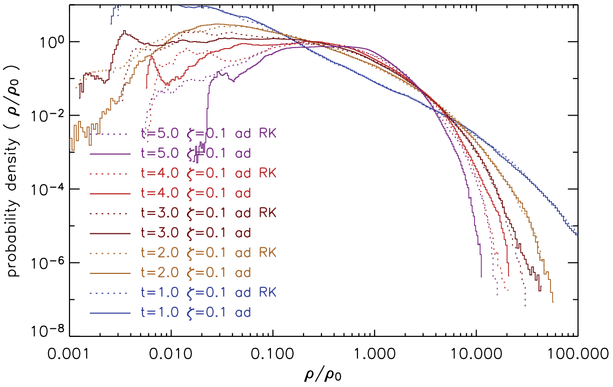


Fig. 31. Time history of mass density PDF for adiabatic runs. The curves are labelled as in Fig. 28. Times are given in units of integral time scale T.

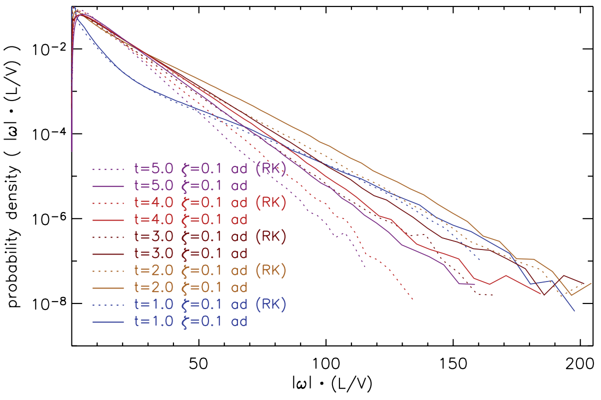


Fig. 32. Time history of $|\omega|$ PDF for adiabatic runs. The curves are labelled as in Fig. 28.

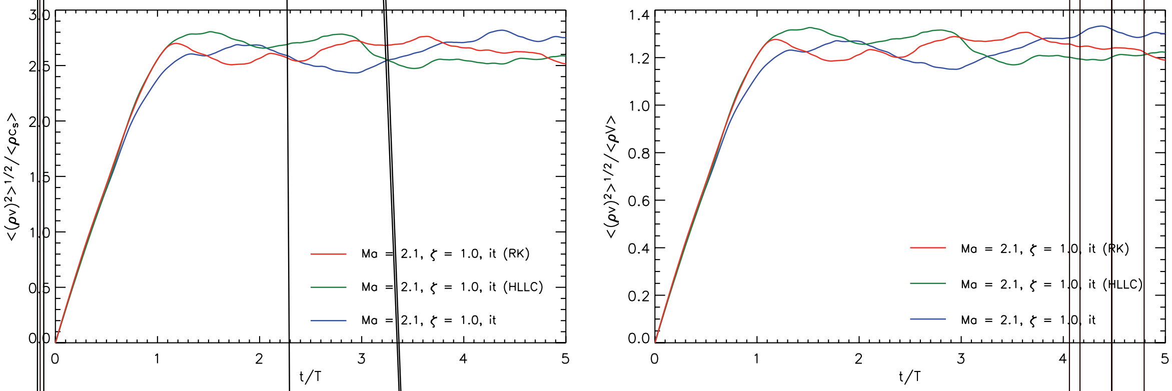
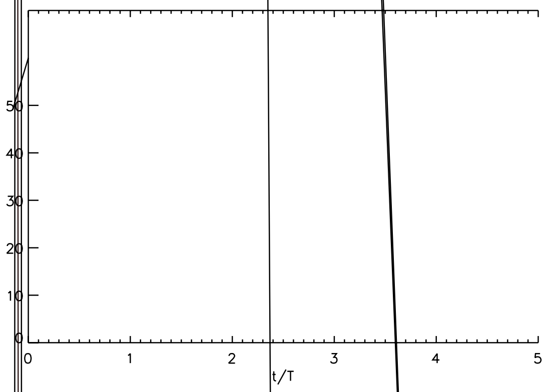


Fig. 33. Time history of RMS (root mean squared) Mach number and momentum for isothermal run with characteristic Mach number 2.1. The curve from PPM is labelled '*it*', from PPM-HLLC '*it (HLLC)*', and from RK-HLLC '*it (RK)*'.



ising that the RK-HLLC code does not appear comparably more dissipative in these iso-
d in the adiabatic case, and in the one- and two-dimensional tests. For the isothermal test
Mach number 21.1, it is not so unexpected. This is because the RMS Mach number was
n the adiabatic runs, and from Fig. 30 it seems that the difference is less for higher Mach

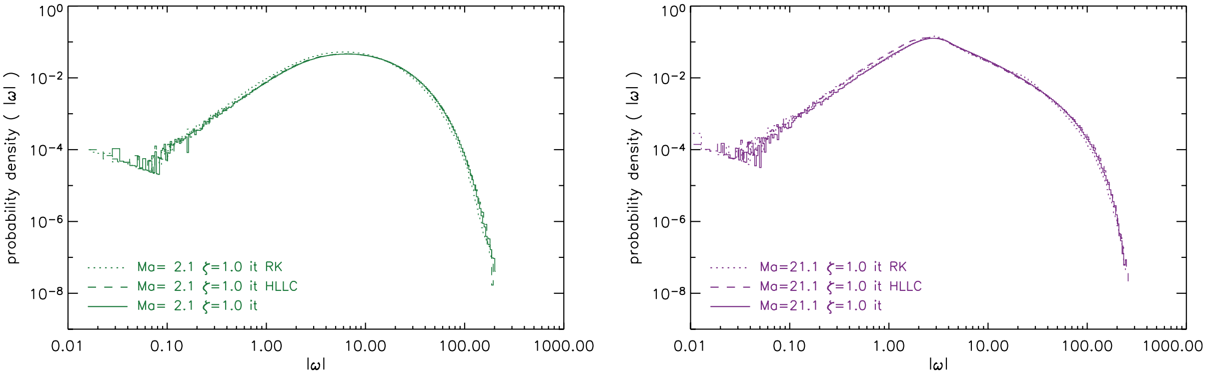


Fig. 36. Vorticity PDFs for isothermal runs from the final time $t = 5T$. On the left Ma = 2.1, and on the right Ma = 21.1. The curves are labelled as in Fig. 33.

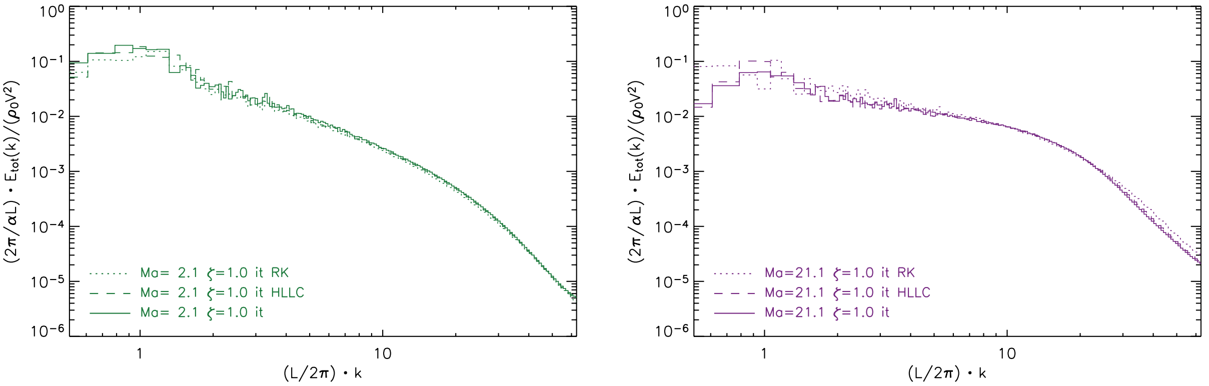


Fig. 37. Energy spectra for isothermal runs from the final time $t = 5T$. On the left Ma = 2.1, and on the right Ma = 21.1. The curves are labelled as in Fig. 33.

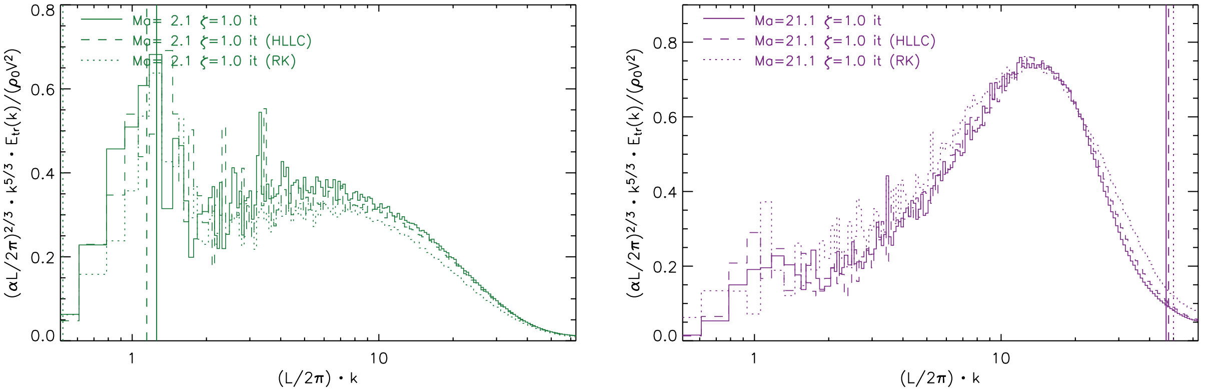


Fig. 38. Compensated energy spectra for isothermal runs from the final time $t = 5T$. On the left Ma = 2.1, and on the right Ma = 21.1. The vertical lines represent the 'sonic wave number'. Integrating the curve to the left of his line gives c_s^2. The curves are labelled as in Fig. 33.

numbers. For the test with characteristic Mach number 2.1, however, the RMS Mach number is comparable to that of the adiabatic tests at around $t = 3T$. Hence, in addition to Mach number, the equation of state must be taken into account when comparing RK-HLLC and PPM. In Section 2, we noted that the most significant difference between the RK- and PPM-algorithms was that both RK-codes smeared out contact discontinuities more. In the isothermal case contact waves are not present, which might explain why the codes differ less here than in the adiabatic runs.

5. Summary

From our numerical experiments we have made the following observations:

- There is slightly more smearing of stationary shocks with HLLC–Bouchut compared to the exact solver.
- The RK-codes smear out most features more than the PPM-codes, and especially contact discontinuities.
- RK-HLLC handles small densities better than the other codes.
- All codes exhibit spurious oscillations. We see more of them with the PPM-codes, except at the near stationary shock, where PPM has a specialised 'flattening' procedure.
- The growth of Kelvin–Helmholtz and Richtmeyer–Meshkov instabilities appears to be little affected by which of the two Riemann solvers are used.
- In turbulence simulations of adiabatic gas, the dissipativity of RK-HLLC seems to be less for higher than for lower Mach numbers, while the dissipation with PPM is independent of the Mach number.
- For turbulence in an adiabatic gas with an RMS (root mean squared) Mach number less than about 5, RK-HLLC seems to be more dissipative than PPM.
- For turbulence with an RMS (root mean squared) Mach number of 2.5 and higher in an isothermal gas, there were no significant differences between Riemann solvers or higher algorithms.

The widespread use of PPM in the astrophysics community has lead to concern about how much the results depend on this algorithm. We conclude that with respect to the Riemann solver their results are accurate. However, the efficiency of the HLLC Riemann solver of Bouchut suggests that it may be used instead.

Acknowledgments

The authors would like to thank Prof. Jens C. Niemeyer at the University of Würzburg for valuable comments and suggestions. We also thank Christoph Federrath for performing the post-processing of the turbulence simulation data.

References

[1] François Bouchut, Entropy satisfying flux vector splittings and kinetic BGK models, Numer. Math. 94 (4) (2003) 623–672.

[2] François Bouchut, Nonlinear stability of finite volume methods for hyperbolic conservation laws and well-balanced schemes for sources, Frontiers in Mathematics, Birkhäuser Verlag, Basel, 2004.

[3] Phillip Colella, Harland M. Glaz, Efficient solution algorithms for the Riemann problem for real gases, J. Comput. Phys. 59 (1985) 264–289.

[4] Phillip Colella, Paul R. Woodward, The piecewise parabolic method (PPM) for gas-dynamical simulations, J. Comput. Phys. 54 (1984) 174–201.

[5] Frédéric Coquel, Philippe G. LeFloch, A second order entropy satisfying scheme for systems of conservation laws (Un schéma entropique du second ordre pour les systèmes de lois de conservation), C.R. Acad. Sci., Paris, Sér. I 320 (10) (1995) 1263–1268.

[6] Wolfgang Dobler, Nils Erland L. Haugen, Tarek A. Yousef, Axel Brandenburg, Bottleneck effect in three-dimensional turbulence simulations, Phys. Rev. E 68 (2003) 026304.

[7] Uriel Frisch, Turbulence, Cambridge University Press, Cambridge, 1995.

[8] B.A. Fryxell, E. Müller, W.D. Arnett, Hydrodynamics and nuclear burning, MPA preprint, 449, 1989.

[9] Amiram Harten, Peter D. Lax, Bram van Leer, On upstream differencing and Godunov-type schemes for hyperbolic conservation laws, SIAM Rev. 25 (1983) 35–61.

[10] W. Hillebrandt, M. Reinecke, W. Schmidt, F.K. Röpke, C. Travaglio, J.C. Niemeyer, Simulations of turbulent thermonuclear burning in type ia supernovae, in: Warnecke Gerald (Ed.), Analysis and Numerics for Conservation Laws, Springer, Berlin, 2005, pp. 363–384.

[11] Jeffrey A. Pedelty, R. Woodward Paul, Numerical simulations of the nonlinear kink modes in linearly stable supersonic slip surfaces, J. Fluid Mech. 225 (1991) 101–120.

[12] M. Reinecke, W. Hillebrandt, J.C. Niemeyer, Thermonuclear explosions of chandrasekhar-mass c + o white dwarfs, Astron. Astrophys. 347 (1999) 739–747.

[13] M. Reinecke, W. Hillebrandt, J.C. Niemeyer, Three-dimensional simulations of type ia supernovae, Astron. Astrophys. 391 (2002) 1167–1172.

[14] Philip L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comput. Phys. 43 (1981) 357–372.

[15] W. Schmidt, J.C. Niemeyer, Thermonuclear supernova simulations with stochastic ignition, Astron. Astrophys. 446 (2006) 627–633.

[16] Wolfram Schmidt, Turbulent thermonuclear combustion in degenerate stars, Ph.D. Thesis, 2004.

[17] Wolfram Schmidt, Christoph Federrath, A parameter study of forced supersonic turbulence in large eddy simulations, in press.

[18] Wolfram Schmidt, Wolfgang Hillebrandt, Jens C. Niemeyer, Numerical dissipation and the bottleneck effect in simulations of compressible isotropic turbulence models, Comp. Fluids 35 (2006) 353–371.

[19] Igor V. Sytine, David H. Porter, Paul R. Woodward, Stephen W. Hodson, Karl-Heinz Winkler, Convergence tests for the piecewise parabolic method and Navier–Stokes solutions for homogeneous compressible turbulence, J. Comput. Phys. 158 (2) (2000) 225–238.

[20] E.F. Toro, M. Spruce, W. Speares, Restoration of the contact surface in the HLL-Riemann solver, Shock Waves 4 (1) (1994) 25–34.

[21] Eleuterio F. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, second ed., Springer-Verlag, Berlin, 1999 (a practical introduction).