High-resolution SPH simulations of a 2D dam-break flow against a vertical wall. Commentary to the collection of datasets

Giordano Lipari* and Cornelis (Kees) Vuik†

Numerical Analysis Group, Delft Institute for Applied Mathematics, Delft University of Technology - The Netherlands

Version 2: December 16, 2021‡

Abstract

This collection of datasets may appeal to those interested in computer simulations of water flows with a free-surface using Smoothed Particle Hydrodynamics. Weakly compressible SPH is a computationally intensive and computationally demanding technique. Obtaining high resolution datasets can be either difficult, impractical or unaffordable for many scholars and practitioners.

The datasets are simulations at four spatial resolutions of the benchmark solution of a 2D dam break impinging on a vertical wall used by D.D. Meringolo, S. Marrone, A. Colagrossi and Y. Liu — see "A dynamic δ-SPH model: how to get rid of diffusive parameter tuning" in *Computers and Fluids*, 2019, 179:334-355, at https://doi.org/10.1016/j.compfluid.2018.11.012.

The resolutions of the initial water column with 800, 1600, 3200 and 6400 particles have led to a total fluid-particle count between 1M and 82M. The simulations have been run with the open-source solver DualSPHysics 5.0 on a GPU device with a nominal peak throughput of 5300 GFLOPS in double precision. In line with the policy of the Delft University of Technology to produce FAIR research output — 'fair' for findable, accessible, interoperable, reusable —, the 4TU Centre for Research Data archives these SPH simulations in a dataset collection retrievable from https://doi.org/10.4121/c.5353691 under a CC-BY 4.0 license.

Possible uses are severalfold. This collection provides data on an established benchmark usable, for example, for education and dissemination; the in-depth visualisation and analysis of results; the execution of refined or extended simulations; the comparison of behaviours and results against changes of parameter settings and across different solvers; convergence and sensitivity studies.

Keywords: Smoothed Particle Hydrodynamics, dam-break flow, GPU computing, DualSPHysics solver, numerical experiment, benchmark datasets

*https://orcid.org/0000-0002-5680-2809, e-mail: glnl@ymail.com. Formerly postdoctoral researcher.

†https://orcid.org/0000-0002-5988-9153, e-mail: c.vuik@tudelft.nl.

‡Begun on 15 Dec 2020. Version 1: 30 March 2021.

Contents

1 Generalities

1.1 Briefly on physics

Broadly speaking, the property of fluidity consists in the fact that small portions of a body of liquid easily move with velocities different to their neighbours; at least, this variability in time and space is distinctive of fluidity. Computer simulations of this behaviour use two principal approaches; intermediate approaches do exist, but are not essential here.

1. The Eulerian approach first divides the space occupied by the body of liquid into a large number of small volumes *fixed in space*; and then, as time unfolds, tracks the properties of the liquid flowing across these fixed conceptual volumes.
2. The Lagrangian approach first divides the body of liquid itself into a large number of small masses that are *not fixed*; and then tracks the flow of each of these masses as they travel around. Lagrangian approaches are most suited to water flows with fragmented free surfaces. These are often associated to rapid and violent dynamics, such as wave breaking and impacts.

The Smoothed Particle Hydrodynamics (SPH) methodology does use a Lagrangian approach and is suitable for modelling rapid and violent flows.

1.2 Briefly on modelling

SPH has originally been developed in astrophysics as a modelling methodology for truly compressible matter. However, in most engineering and almost all environmental applications, water is very nearly incompressible. A substantial strand of SPH research has then devoted itself to adapting the original SPH formulation to modelling water as a weakly compressible fluid.

Even more precisely, the compressibility in weakly compressible SPH takes artificial values defined by the modellers. This artificial tweak does not impair greatly the correct simulation of a genuine nearly incompressible fluid; the proviso is not to draw specific conclusions on processes crucially dependent on real compressibility, such as the propagation of sound.

This document does not expose the tenets of SPH. Please see the review paper of Monaghan (2012) as an entry point. It is important for the sequel to recall that an SPH particle's state at a time instant consists of:

- position, after the Lagrangian formulation;
- velocity, after the fundamental focus on fluid flow simulation;
- density, after the weakly compressible formulation.

1.3 Briefly on the solver

DualSPHysics:

1. is an SPH solver with a wide range of modelling capabilities for engineering and environmental applications, implementing an artificial weak compressibility. The code can compile and run on CPU and GPU devices.
2. has been developed by a team of developers and contributors from academic and research institutions, among others the Environmental Physics Laboratory of the Universidade de Vigo in Spain and the School of Mechanical, Aerospace, and Civil Engineering of the University of Manchester in the United Kingdom.

   The website https://dual.sphysics.org/ is the entry point for the full list of developers; features, documentation and references; downloads and user interface; FAQ, forum and news; training and tutorials; animations and visualization.
3. is a suite of programs in which:
   (a) DualSPHysics proper is free software distributed under a GNU General Public License v2.1 and BSD license. It uses other pieces of free software adding functionalities to it;
   (b) closed-source utilities are provided for the pre-processing and post-processing operations wrapped around the processing proper — more in § 3.2.

The two pathways for accessing the software suite are not entirely equivalent:

1. The GitHub repository at https://github.com/DualSPHysics/DualSPHysics contains branches up-to-date with the continuous code development. However, the documentation contains only two example cases and no detailed users' guides.
2. A zipped archive from https://dual.sphysics.org/index.php/downloads/, last accessed 13 Dec 2021, contains the code frozen at the latest tagged version and the complete documentation and set of examples.

Note that the versions available to the general public are not necessarily those used to produce scientific advances published elsewhere.

Resources guiding the users through the workflow of DualSPHysics are:

1. The GitHub wiki page: https://github.com/DualSPHysics/DualSPHysics/wiki, last accessed 13 Dec 2021. This is the principal source of the information on DualSPHysics reported in this document;
2. The help documentation of the software utilities (in the repository and archive);
3. The templates for the XML input files (in the repository and archive);
4. The users' guides in PDF format (in the archive only).

1.3.1 Disclaimers

- Please always double check the original sources of DualSPHysics as a safeguard against possible errors and omissions in this document, beside the fact that they provide the official updates and releases.
- Please take note that the Delft Institute of Applied Mathematics is not a member of the development team of DualSPHysics.

  In fulfilment of the licensing terms, the information provided with this collection of datasets and with this accompanying document implies neither endorsement nor promotion of DualSPHysics. Likewise, citation of the names of DualSPHysics contributors implies neither their endorsement nor their promotion of this collection of datasets and of this accompanying document.

2 The dam-break flow impinging on a vertical wall

2.1 As an experiment

In a dam-break flow a boxful of water is released inside a tank upon removing one side of the box. The water runs against the opposite side of the tank, hits the end wall and splashes up and back again. This flow is traditionally called 'dam break' because releasing suddenly a mass of still water associates with the failure of a dam.

Broadly speaking, the flow evolution consists of four stages:

1. As the water box collapses and the leading edge advances over the box floor, the flow is gradually varied;
2. Then follows the impact of water against the wall, hence an instance of fluid-structure interaction;
3. Then, the water raising at the wall plunges back on the upstream water, and water-water impacts cause intense fragmentation of the free surface, droplets, bubbles and vortical motions in the bulk of the fluid;
4. Finally, the much-agitated water begins to slosh and settle into a quiescent state thanks to the action of viscosity.

The physics of the last two stages is highly nonlinear and is amenable to an SPH-based solving approach.

A dam-break experiment can be material or numerical. This collection contains reproductions of the *dam-break flow impinging at a vertical wall* provided as a benchmark numerical solution by D.D. Meringolo, S. Marrone, A. Colagrossi and Y. Liu — see "A dynamic δ-SPH model: how to get rid of diffusive parameter tuning" in Computers and Fluids, 2019, 179:334-355 by Meringolo et al. The paper itself points to past research developed with that long-standing and widely used benchmark.

In that numerical experiment, air is not modelled and the flow is two-dimensional. The problem is solved in dimensionless form, with a few noteworthy consequences:

- The problem variables describing the geometry, kinematics and dynamics express ratios of dimensional quantities to the height of the water box (H, whose value may remain unstated), the acceleration of gravity g, and the reference fluid density, ρ_0;
- Also the computed time and velocity express ratios to scales derived from the reference values above. The scales for time and velocity, $\sqrt{H/g}$ and $\sqrt{gH}$, are formally identical to the parameters of the forward propagation of the leading edge of the collapsing water column, as in the flow stage 1 above. Note, in passing, that the free-fall velocity, $\sqrt{2gH}$, relates with the motion of individual water droplets more closely;
- All simulation parameters with a physical dimension, for example the artificial speed of sound, are expressed in dimensionless form as well.

Please refer to Meringolo et al. (2019) for more information about the problem geometry.

2.2 As a simulation

DualSPHysics, described in § 1.3, is an SPH solver different to that used by Meringolo et al. (2019). This section outlines the main aspects of the general formulations and specific settings used to produce this collection of datasets; differences of detail between the two source codes are not commented upon. Dr Andrea Colagrossi of CNR-INM (National Research Council, Institute of Marine Engineering, Italy) has kindly shared the original simulation settings.

2.2.1 SPH governing equations

- The kernel function is a Wendland C2 formulation for both simulations: its characteristic smoothing length, h, is half the radius of the support. The radius of the support is also referred to as kernel size or cut-off length. In simulations the smoothing length h, an SPH parameter, is expressed as a multiple of the interparticle distance in the initial conditions, Δp, a discretisation parameter: commonly, $h/\Delta p$=2.
- As for the equation of state, both solvers implement a linear relationship between density and pressure; the pressure is relative to the total pressure for which the density takes the reference value ρ_0. The Mach number of the simulation is 0.1, since the dam-break celerity is always 1 unit and the artificial speed of sound takes the value of 10 units; this is well in the range of weakly compressible behaviour. Density can be converted into pressure with this equation of state.
- As for the mass-conservation statement and the density-diffusion term characteristic of SPH, the DualSPHysics simulation adopts a formulation by Molteni and Colagrossi (2009), while the simulation by Meringolo *et al.*, 2019 adopts a δ-SPH formulation invoking Randles and Libersky (1996).
- As for the momentum-conservation statement, in both simulations the viscous term is expressed with the artificial-viscosity formulation:

$$\alpha h c_0 \frac{\rho_0}{\rho_i} \sum_j \Pi_{ij} \nabla_i W_{ij} V_j$$

The source code of DualSPHysics has been modified in order to substitute the formulation of the function Π_{ij} by Monaghan and Gingold (1983) — devised to alleviate specific astrophysical problems such as colliding gas clouds — with the same formulation as Meringolo *et al.*, 2019:

$$\Pi_{ij} = \frac{(\vec{u}_j - \vec{u}_i)(\vec{r}_j - \vec{r}_i)}{(\vec{r}_i - \vec{r}_j)^2}.$$

See § 3.2.1 for the implementation of this modification.

Meringolo *et al.*, 2019 resolved a dam-break problem with a range of the artificial viscosity parameter, α; in this dataset, $\alpha = 0.01$ only.

- Finally, the boundary conditions prescribe free slip at solid-fluid interfaces. In DualSPHysics this has been implemented by means of the so-called 'dynamic boundary conditions': gauged repulsive forces exerted by the in-boundary particles keep the free-flowing particles 1.5 smoothing lengths away from the boundary surface (Crespo et al., 2007). Meringolo *et al.*, 2019 use the established technique of 'ghost particles' whereby the fluid particles can approach the boundary surface freely.

2.2.2 Numerical parameters

- The smoothing length is twice the initial interparticle distance, $h/\Delta p = 2$, in both simulations.
- The $H/\Delta p$ ratios discussed in Meringolo *et al.*, 2019 vary between 480 and 1600. The present collection contains four reproductions of the same benchmark with spatial-resolution ratios of 800, 1600, 3200 and 6400.

  For the given geometry, the counts of fluid particles are 1,280, 5,120, 20,480 and 81,920 thousands respectively.
- The time integration method is a Runge-Kutta formulation with a frozen diffusive approach in Meringolo *et al.*, 2019; and a single-stage, two-step, 'kick-drift' Verlet scheme in DualSPHysics.

  Both methods march in time explicitly and are subject to a stability constraint: in both cases the Courant number is 0.125. Note that the time step for the entire cloud of particles is reduced based on the stability of the particle with the highest velocity of all.
- The simulated time is 20 units: at that point in time, the energy dissipated by the system is estimated at around 88% of the initial energy — see Fig. 16 in Meringolo et al. (2019). This is the early stage 4 mentioned in § 2.1.

Note that a higher resolution in SPH simulations entails a considerable compute workload. A higher number of particles increases the operations at each time instant. Also, the ratio of the smoothing length to the artificial speed of sound, h/c, is a scale for the initial time step. Owing to the stability of explicit time-marching methods, the time steps are smaller for smaller spacings and the temporal resolution increases accordingly. Finer spatial and temporal scales cause a wider range of velocities to be resolved, thus further occasions for reductions of the time step. Therefore, simulations at an increased spatial resolution do take longer to complete. Table 1 summarises these trends for the datasets.

Table 1: Numerical parameters and problem size in the collection. $h/\Delta p = 2$.

$H/\Delta p$	800	1600	3200	6400
h/H	0.00250	0.00125	0.000625	0.0003125
Fluid particles (10^3)	1,280	5,120	20,480	81,920
Time steps	1,395,921	2,863,036	5,728,950	11,743,221

3 The collection of simulation datasets

3.1 Usages of a shared dataset

Completing a high-resolution SPH simulation requires powerful compute devices such as GPUs and, of course, time. The increased workload affects the stages of pre-processing, processing proper and post-processing alike. In sum, high-resolution simulations push the limits of the affordable and practical.

The sharing of this collection has had in mind general aims and specific usages:

- In broad terms, sharing aims to stimulate and facilitate the informed access to general and special features of modelling rapid flows with weakly compressible SPH.

  Consider also that a compute resolution that is at entry level for one field of application can be high for another field of application. Therefore, a particular benchmark can offer insights not limited to the specific subtopic where they originated from (dam-break flows, here).
- In more specific terms, a dataset of pre-computed, highly resolved results lends itself to several usages. Here, some are envisaged in a loose order of increasing complexity:
  1. preparing educational and presentational material;
  2. deriving secondary flow variables from the particles' states; for examples, the fields of pressure and vorticity;
  3. serving as initial condition for hot starts, including refined investigations of the temporal intervals between the output frequency — see § 3.2.4;
  4. serving as a yardstick to anticipate the loss of information incurred at the resolution that users can afford in their own situation;
  5. contrasting the results with practical knowledge and with measurements, taking into account the modelling assumptions;
  6. comparing with the results obtained with other weakly compressible SPH solvers;
  7. studying the sensitivity to physical parameters;
  8. quantifying the changes introduced by new submodels for special physics;
  9. studying the sensitivity to numerical parameters;
  10. quantifying the gains in accuracy and performance that numerical expedients can deliver in the nonlinear flow stages outlined in § 2.1, where conventional convergence metrics have no tractable trend;
  11. analysing the dataset in terms of frequency and distribution of nonlinear events; for example, fragmentations/collapses/impulses.

Post-processing is possible using the closed-source tool kit provided in the DualSPHysics suite — see § 3.2.8. Beside other capabilities, this tool kit produces files that can be read by the open-source ParaView, which has in turn its own data analysis capabilities — see § 3.2.9. Both the native DualSPHysics binaries and the ParaView files are part of the datasets.

Take note that the collection also contains the information needed for replaying the simulations and reproducing the data, as detailed next.

3.2 The creation of the datasets

We give file sizes in bytes and decimal prefixes. So 1 MB is 1000 B and 1 GB is 1000 MB.

3.2.1 DualSPHysics source code

- The version used is DualSPHysics v5.0.164 of 27-11-2020; the corresponding commit can be downloaded from the GitHub repository at https://github.com/DualSPHysics/DualSPHysics/releases/tag/v5.0.164.

- The collection does contain a zipped file of this commit, downloaded on 15 Dec 2020. Importantly, the zipped archive contains the license terms of DualSPHysics — recall § 1.3.
- When unzipped, the archive file of 307 MB expands to a directory tree totalling 786 MB worth of documents. The abridged directory tree is show below; in the following we indicate the root element of this tree as dsph-root.
- The source code is contained in the subdirectory dsph-root/src/source and has been edited in two places, namely for
  1. the implementation of the viscous terms, as explained in § 2.2.1;
  2. the number precision in the output of the 'gauges' numerical device — see § 3.2.4.

  The edits with respect to the DualSPHysics release are listed in a 'patch document' included in the collection — see § 3.3.2. The patch is human-readable list of the files and lines changed, produced with the Git command git diff [original code] [modified code]. The command git apply can be used to reproduce from the original code the modified code. Two entry points for the working of these Git commands are the manual pages at https://git-scm.com/docs/git-diff and https://git-scm.com/docs/git-apply, last accessed 13 Dec 2021.

The abridged directory tree of the DualSPHysics suite, as distributed in the zipped archive, is:

```
DualSPHysics-ec025723b947f9d3af96ead3025393681b4fd1ed [dsph-root]
├── linux
├── windows
├── doc
│   └── ...
├── examples
│   └── ...
└── src
    ├── lib
    │   └── ...
    ├── source
    ├── VS
    └── src_mphase
```

3.2.2 Compilation

- The source code of DualSPHysics has been compiled in a machine using the distribution Debian GNU/Linux 10 (codename "buster") as operating system.
- The compilation tools were g++ 8.3.0 for C++ and the Software Development Kit 10.1, v10.1.168 for CUDA.
- The building process has been driven by GNU Make 4.2.1; a template makefile configuration file is distributed with the DualSPHysics suite.

  That configuration file has been edited to accommodate for the CUDA software and the compute capability of the GPU device available locally; the edited version of the Make configuration file is not part of this collection.

3.2.3 Hardware

The GPU used for computing is a Nvidia Quadro GP100. A white paper with technical specifications is available from https://images.nvidia.com/content/pdf/tesla/whitepaper/pascal-architecture-whitepaper.pdf, last accessed 13 Dec 2021. The nominal peak throughputs are 5.3 GFLOPS in double precision and 10.6 GFLOPS in single precision.

Information on the device is available in the simulation output — see § 3.2.7; however, we recommend not to take the reported compute times as a measure of performance blindly: the runs have been performed while the

machine processed varying workloads, and no analysis to disentangle confounding factors had been carried out at the time of writing this document.

3.2.4 Simulation settings (XML file)

The user states the simulation settings for DualSPHysics in a text file in XML format. The XML file format is open and platform-independent.

- The user-defined XML settings file, Meringolo_Def.xml, is a part of the collection.
  This file can be used to start a fresh simulation as well as to restart a simulation from an intermediate output time. It is strongly recommended to inspect this file for familiarising oneself with limitations and possibilities of the datasets, as for both the physical and the numerical modelling.
  Note that in the simulations the particle-spacing parameter, Δp, has been assigned as a command argument in the driver script, commented next in § 3.2.5, rather than as a statement within this XML file.
- The frequency of the output data is set in this file. Namely:
  - Fields. The particle states are saved to file every 0.10 time units. Since the simulated time is 20 units, the output rounds are 201, including the reading out of the initial conditions. Clearly, the number of output rounds, hence of files, determines the size of a dataset.
    These files are part of the collection — see § 3.3.2 for the data organization.
  - Gauges. These gauges are numerical devices implemented in DualSPHysics that record the time evolution of flow quantities at points in space. The simulations had gauges record the water-surface height at three locations, and the velocity in three arrays of 21 numerical probes placed at the corners and middle of the tank.
    This output is not part of the collection.
  - Time steps. The values of all variable time steps during a simulation have also been recorded.
    This output is not part of the collection.

3.2.5 Driver script

The collection contains the script file that launched the pre/proper/post-processing executables, meringolo.sh.

- This driver script is written for a GNU Bash shell, so specifically for a Unix-like operating system. Nonetheless, the document is shared to show the arguments passed on to the utilities of the DualSPHysics suite.
  Of particular relevance is the fact that DualSPHysics runs with the -stable flag: this option acts on the memory management procedures and produces precisely the same results as a simulation is repeated (information courtesy of https://forums.dual.sphysics.org/discussion/1537/; last accessed 13 Dec 2021). This option is important for replaying the simulations and obtaining the same results.
- This driver script also determines a directory tree for the output files and their names. The tree structure is shared below for illustration, so that the users can interpret the driver script and adapt it to their preferred work-flow. Its root element, simulation-root, is a local path designated in the script. Note, though, that the collection repository does not permit folders, whereby § 3.3.2 will describe the data organisation in the datasets after this output directory tree has been flattened.

The abridged directory tree of the simulation output is:

```
[simulation-root]
```

```
├─ data # see § 3.2.7
├─ gencase # see § 3.2.6
└─ particles # see § 3.2.8
```

3.2.6 Pre-processing output files

The user-defined XML settings file of § 3.2.4 is fed into the pre-processing closed-source utility GenCase v5.0.197 (18-07-2020).

- GenCase is distributed in the archive subdirectory dsph-root/bin/linux shown in § 3.2.1; refer to the subdirectory dsph-root/doc/help for the command-line arguments and options.
- The driver script gathered this output in the subdirectory simulation-root/gencase of the output directory tree, as shown in § 3.2.5.
- The output of GenCase consists of a few ParaView-readable files for the domain design, a log file, and an input XML file for DualSPHysics.

  GenCase generates that *second* XML file (Meringolo.xml) upon appending some pre-processing information to the user-defined file mentioned in § 3.2.4 (Meringolo_Def.xml).

  This second XML file is part of the datasets; it can be used to start a fresh simulation too.

3.2.7 Proper-processing output files

DualSPHysics on roads produces two streams of output files:

1. A log file of the simulation called Run.out.

   It contains important information like the input parameters, the hardware configuration, the domain configuration, and the run-time progress.

   The driver script of § 3.2.5 saved this log file in the root of the output directory tree, simulation-root.

   This file is part of the datasets. We advise not to rely on the compute times reported in this log file but in gross numbers — recall § 3.2.3.

2. Binary files storing the states of the particles at the output frequency prescribed in the XML settings file, as outlined in § 3.2.4.

   These binary files have format extensions bi4, obi4, ibi4. The resources of § 1.3 do not describe their file format specifications; a developer can reverse-engineer these format specifications from the classes on input/output in the DualSPHysics open-source code — see for example https://forums.dual.sphysics.org/discussion/comment/2306, last accessed 13 Dec 2021.

   These binary files are important for restarting a simulation from the flow state at a chosen output time ('hot start').

   The driver script of § 3.2.5 saved these files in the output subdirectory simulation-root/data.

   These files are part of the datasets. Each dataset contains the pair PartInfo.ibi4 and Part_Head.ibi4, one PartOut_0000.obi4 file, and 201 Part_????.bi4 files, where the question marks indicate a frame number from 0000 to 0200. This implies 204 binary files per simulation and 816 files in the datasets of four simulations.

   The typical size of a bi4 file for 82M particles is 3.6 GB.

3.2.8 Post-processing output files

The post-processing closed-source tool 'PartVTK v5.0.122 (09-07-2020)' has been used to extract from the binary files of § 3.2.7 information readable with ParaView.

- PartVTK is distributed in the archive subdirectory dsph-root/bin of § 3.2.1; refer to the subdirectory dsph-root/doc/help for the command-line arguments and options.
- With the settings given in § 3.2.5, PartVTK produces as many files in vtk format as there are bi4 files. These vtk files contain the states of the particles at the output times of § 3.2.4; these files are the input for ParaView as outlined next in § 3.2.9.

  These files are part of the datasets. They are named with the pattern WaterParticles_????.vtk where the question marks indicate a frame number from 0000 to 0200. This implies 201 vtk files per simulation and 804 files in the datasets of four simulations.

  The largest vtk file in the collection is 3.3 GB.

3.2.9 Visualisation

Visualisations of the flow field are also part of the collection, in order to help users appreciate the overall development of the dam break and identify the flow stages that are most interesting for them.

- These images have been rendered with ParaView, "an open-source multiple-platform application for interactive, scientific visualization" (https://en.wikipedia.org/wiki/ParaView, last accessed 13 Dec 2021). ParaView can also be used for the data processing of the vtk files.
- The temporal evolutions of the density, particle identification number and velocity fields each output time in each dataset simulation are shown as snapshots in PNG format — see Figure 1. 1604 files are gathered in 12 archive files, one for each simulation and field — see § 3.3.2 for their the archive format.
- A playlist with the animations of the density, particle identification number and velocity fields at the four resolutions is available on YouTube at https://www.youtube.com/playlist?list=PLb_klyJ6w5QihDlztSqN0GRhT7awNnibe, last accessed 13 Dec 2021.

Recall that ParaView will draw each data point individually: a simulation with 82M particles entails a considerable visualisation task, in terms of both memory and compute power. Therefore, we have accelerated visualisation with a client/server arrangement of ParaView 5.7.1, using the GPU used for computing as graphics server. To this end it has been necessary to enhance ParaView with the NVIDIA IndeX™ plug-in. Explaining this is beyond the scope of this document: see https://www.nvidia.com/en-us/data-center/index-paraview-plugin/, last accessed 13 Dec 2021, for an entry point.

3.3 Collection access

3.3.1 Compression

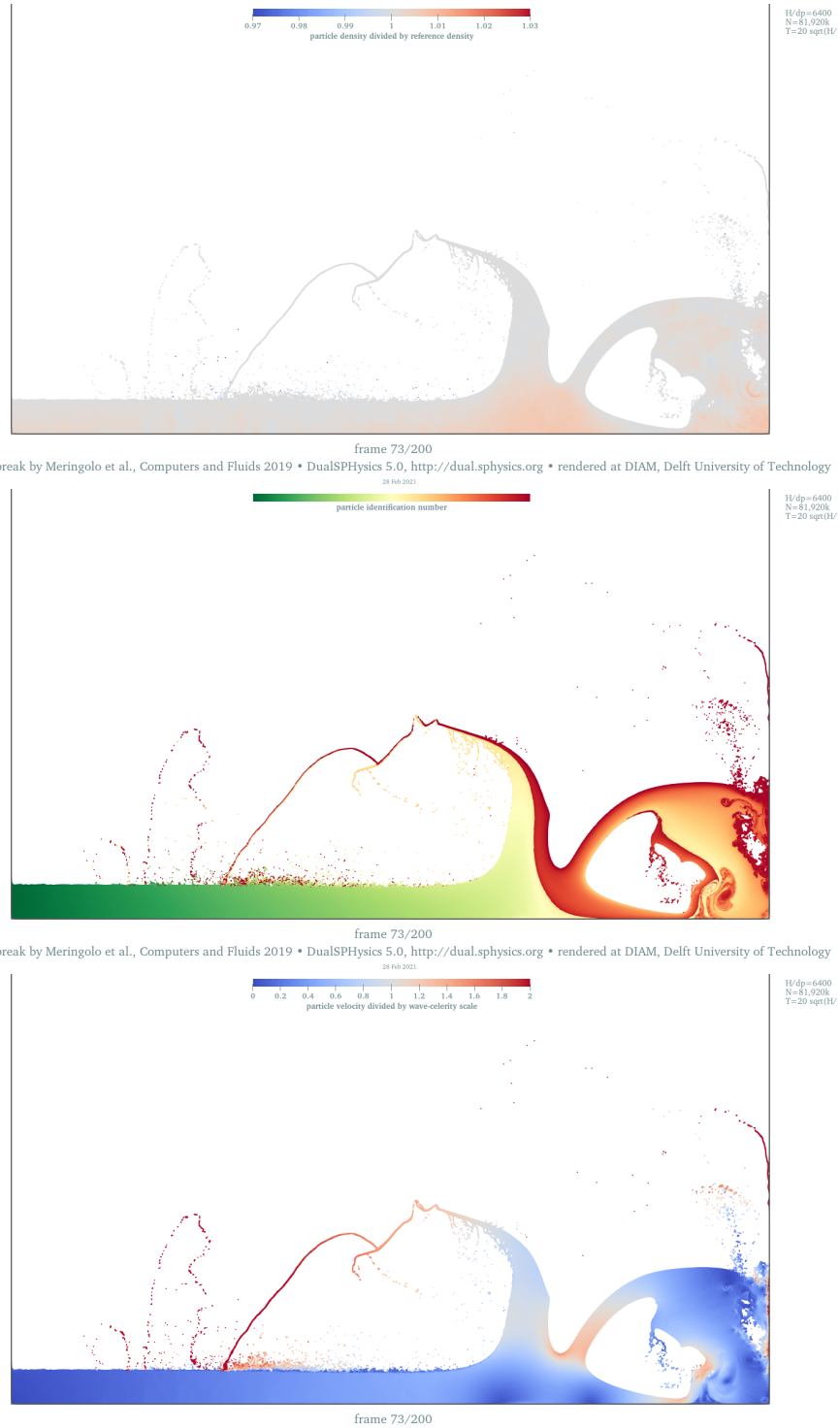
The datasets contain compressed files, save a few exceptions.

Files have been compressed using the xz compression algorithm and file format. Compression with xz is lossless, is primarily based on the compression algorithm LZMA2, has open file-format specifications, and can be handled by cross-platform software. See http://tukaani.org/xz/, last accessed 13 Dec 2021, for details on xz.

The Linux utility used for compressing this dataset is xz (xz Utils) 5.2.4, using the library liblzma 5.2.4. This utility allows multithreading. The default compression level 6 is deemed adequate to strike a compromise between the eventual decompression speed and the memory requirements in computers of varying level. Also, a guarantee of integrity has been enforced using the default CRC64 check type: the compressed file includes a file checksum that is verified by the decompresser program. The GNU Bash command and options summarising the above operation was xz -T0 -6 -C crc64 [arguments omitted].

The eventual file compression rate is around 2 for binary files and 3 for vtk files, depending on the native size. Table 2 summarises the spatial resolution, number of fluid particles, and the sizes of the datasets at each resolution;

Figure 1: Example of flow field images distributed with the collection: density (top), particle identification number (middle), velocity (bottom).



the count does not include visualisation files.[1]

The compressed files have an xz extension appended to the original file name.

Table 2: Total sizes in GB before and after compression for the dataset of each simulation. Figures are rounded.

$H/\Delta p$	800	1600	3200	6400
Fluid particles (10^3)	1,280	5,120	20,480	81,920
Dataset ID	[4]	[3]	[2]	[1]
Native size	11	88	350	1,399
Compressed size	9	36	140	549
Compression ratio	1.24	2.44	2.49	2.55

3.3.2 Collection organisation and dataset inventory

Generally, compressed files are bundled into archive files to simplify their download, and archiving utilities often take care of compression and decompression under the covers. However, Table 2 shows that the overall size of a compressed dataset is still considerable: downloading it in one or even few blocks may be prohibitively heavy and long for many users. Hence, we have opted for a mixed archiving strategy.

The simulation results are released publicly as a collection of five datasets, accessible from https://doi.org/10.4121/c.5353691.

The five datasets are numbered as follows:

[0] A 'main' dataset with referrer https://doi.org/10.4121/14309240, containing

- (a) A human readable README file;
- (b) This one document, Commentary.pdf;
- (c) The compressed zipped archive of DualSPHysics; recall § 3.2.1;
- (d) The 'patch file' Meringolo.patch containing the changes to the source code — recall § 3.2.1.
- (e) The input XML file Meringolo_Def.def; recall § 3.2.4.
- (f) The driver script meringolo.sh; recall § 3.2.5.
- (g) The images of the fields of velocity, density and particle identification number at each output time for each simulation; recall § 3.2.9 and see Figure 1.
  These frames are uncompressed and bundled in a 7z archive; see § 3.3.5 on how to handle this format.

This dataset is meant as an entry point to help the users find out in the ensuing datasets what is most useful for them. It also contains the files to replay the simulations.

[1] The dataset for the simulation $H/\Delta p$=6400, as in Tables 1 and 2, with referrer https://doi.org/10.4121/14309234. It contains

- (a) A human readable README file;
- (b) The preprocessing output of § 3.2.6, Meringolo.xml;
- (c) The log file of § 3.2.7, Run.out;
- (d) The binary files of § 3.2.7, recognizable by the name pattern Part*;
- (e) The vtk files of § 3.2.8, recognizable by the name pattern WaterParticles*;

[2] Ditto for the simulation $H/\Delta p$=3200 with referrer https://doi.org/10.4121/14309171;

[1] At the time of uploading this collection we observed that the online repository of 4TU.ResearchData reported the dataset size value in binary form (GiB) and the unit in decimal form (GB). The values of Table 2 are expressed consistently in GB, so they appear to be larger than shown in the online repository. For context on the file-size units see https://physics.nist.gov/cuu/Units/binary.html, last accessed 13 Dec 2021.

[3] Ditto for the simulation $H/\Delta p$=1600 with referrer https://doi.org/10.4121/14309081;

[4] Ditto for the simulation $H/\Delta p$=800 with referrer https://doi.org/10.4121/14308883.

3.3.3 Licensing

The collection and its datasets are distributed by the 4TU.ResearchData repository under a CC-BY 4.0 license: https://creativecommons.org/licenses/by/4.0/. The referrers to access the dataset have been given in § 3.3.2.

For the licensing information of DualSPHysics, recall § 1.3 and § 3.2.1.

3.3.4 Downloading

The largest compressed files are 1.8 GB: the estimated times to download this are 46, 23 and 5 minutes with connection speeds of 5, 10, 50 Mbps, excluding any other network traffic overhead. The speeds above are lower bounds for DSL, cable and fibre connections respectively.

3.3.5 Decompression

The dataset files can be decompressed in several ways apart from using the xz utility in Unix-like systems:

- Among open and free software, 7-zip works on a wide range of Windows operating systems and is described in detail at https://7zip.org/, last accessed 13 Dec 2021.

  This archiver software has also been ported into Linux as the p7zip package (https://sourceforge.net/projects/p7zip/, last accessed 13 Dec 2021); this may already be available from the package repositories of Linux distributions, for example Ubuntu.

  Conveniently, 7-zip allows multithreading.
- Commercial utilities are generally able to handle xz formats. An example is WinZip, available for Windows and Mac operating systems: see https://www.winzip.com/en/learn/file-formats/xz/, last accessed 13 Dec 2021.

The same utilities are able to extract the content of 7z archives.

4 Acknowledgements

The production of this collection of datasets has been stimulated by the project "GPU acceleration and numerical optimization for SPH" funded by Shell India Markets Private Limited and by the Delft Institute of Applied Mathematics at the Delft University of Technology, The Netherlands. These data and methodology are not intellectual property of Shell.

Dr Andrea Colagrossi of CNR-INM (National Research Council, Institute of Marine Engineering, Italy) has kindly shared the simulation settings of Meringolo *et al.*, 2019. Dr Santosh Illamparuthi, Data Steward at the Faculty of EEMCS, gave helpful directions on the data-sharing process. Jan van der Heul and Egbert Gramsbergen of the 4TU.ResearchData repository provided invaluable assistance in the creation of this collection.

Prof Kees Vuik of the Delft Institute of Applied Mathematics at the Delft University of Technology has supervised the research as Principal Investigator and approved of the release of this document and of the collection of datasets.

References

Crespo, A., Gómez-Gesteira, M., and Dalrymple, R. (2007). Boundary Conditions Generated by Dynamic Particles in SPH Methods. *Computers, Materials & Continua*, 5(3):173–184.

Meringolo, D., Marrone, S., Colagrossi, A., and Liu, Y. (2019). A dynamic delta-SPH model: How to get rid of diffusive parameter tuning. *Computers & Fluids*, 179:334–355.

Molteni, D. and Colagrossi, A. (2009). A simple procedure to improve the pressure evaluation in hydrodynamic context using the SPH. *Computer Physics Communications*, 180(6):861–872.

Monaghan, J. (2012). Smoothed Particle Hydrodynamics and Its Diverse Applications. *Annual Review of Fluid Mechanics*, 44(1):323–346.

Monaghan, J. and Gingold, R. (1983). Shock simulation by the particle method SPH. *Journal of Computational Physics*, 52(2):374–389.

Randles, P. W. and Libersky, L. D. (1996). Smoothed Particle Hydrodynamics: Some recent improvements and applications. *Computer Methods in Applied Mechanics and Engineering*, 139:375–408.