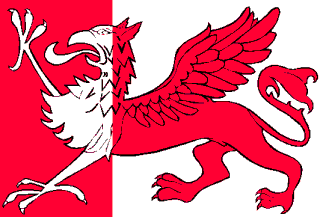
*******Stellingwerf Consulting***

*Complex Systems Analysis*

*Huntsville, AL*

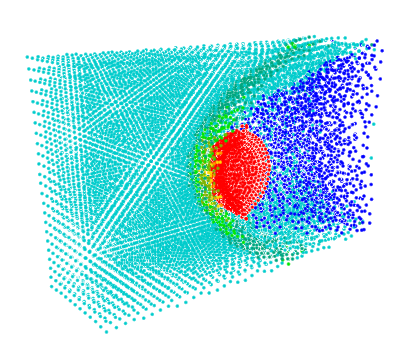
SPHC

Smooth Particle Hydrodynamics Code

Validation Test Suite

3/01/2020

R. F. Stellingwerf





**STELLINGWERF CONSULTING**

11033 Mathis Mountain Road SE, Huntsville, AL 35803

256-880-9789 rfs@stellingwerf.com

**©2020 Stellingwerf Consulting**

**Table of Contents**

Cover Picture – frame shot from the Drop-3D test case, showing a spherical water drop interacting with a Mach 5 air flow.

Colored on density.

[1. Introduction 4](#_Toc48894966)

[2. Test Implementation 5](#_Toc48894967)

[3. Test Descriptions and Results 9](#_Toc48894968)

[a. Shock Tube – 1D 9](#_Toc48894969)

[b. Noh Shock – 1D 12](#_Toc48894970)

[c. Blast Wave - 1D 14](#_Toc48894971)

[d. Blast Wave – 2D 17](#_Toc48894972)

[e. Flyer Plate – 1D 20](#_Toc48894973)

[f. Flyer Plate – 3D 22](#_Toc48894974)

[g. Rotating Rod – 2D 24](#_Toc48894975)

[h. Rotating Cone – 2D 26](#_Toc48894976)

[i. Cube Impact – 2D 27](#_Toc48894977)

[j. Cylinder Fracture – 3D 29](#_Toc48894978)

[k. Tennis Ball – 2D 31](#_Toc48894979)

[l. Solar Panel Impact – 2D 32](#_Toc48894980)

[m. Ball on Plate Impact – 2D 34](#_Toc48894981)

[n. Plate and Bulkhead Impact – 2D 37](#_Toc48894982)

[o. Water Drop in Mach 5 Wind Tunnel – 2D 41](#_Toc48894983)

[p. Water Drop / Rectangular Mach 5 Wind Tunnel– 3D 45](#_Toc48894984)

[q. Water Drop / Cylindrical Mach 5 Wind Tunnel – 3D 49](#_Toc48894985)

[r. Flyer Plate Impact – 1D 53](#_Toc48894986)

[s. Flyer Plate Impact – 3D 55](#_Toc48894987)

[t. Stiff Foam Cylinder Impact – 2D 58](#_Toc48894988)

[u. Breaking Water Wave – 2D 61](#_Toc48894989)

[v. Flow from a Pipe - 3D 65](#_Toc48894990)

[w. Units Conversions File 67](#_Toc48894991)

[4. Applications 70](#_Toc48894992)

[a. Explosion 70](#_Toc48894993)

[b. Debris Cloud 73](#_Toc48894994)

[c. Woven Structures 76](#_Toc48894995)

[d. Folded Structures 81](#_Toc48894996)

[5. References 87](#_Toc48894997)

[6. End 95](#_Toc48894998)

## 

## Introduction

This document describes the test cases used to validate the hydrocode SPHC. These cases serve two purposes – 1) Code validation to ensure that updates do not change any fundamental code results, and 2) Templates for initial setup of SPHC applications. Normally, the first purpose requires that all the tests run normally and produce correct results whenever anything changes in the code structure or run environment. The second purpose provides a tested, working starting point for any application that includes any special setup commands that may be overlooked in a scratch setup for the case. All the test cases use a very small particle number and make use of other time saving setup options to produce fast running test cases. Also, these test cases do not use mature physics models for material properties, equation of state, etc. They are meant to be used as tests *ONLY*, not as actual models of any physical system.

These tests are run whenever the code changes in any way, including coding changes, compiler changes or platform change. At present there are 22 tests in the suite. These tests cover many, but not all, of the physics options, materials, geometry options, boundary conditions, and special features. Many of the tests are standard cases with known, possibly analytic, results. Other tests have been compared in detail with experimental results. Several of these tests are difficult cases to compute with a smooth particle code, and require careful choice of parameters, setup conditions, and boundaries. For these cases, the appropriate case in this list can be used as a template for a successful model of a similar application case.

A partial list of some of the fundamental SPH research papers is given in [Section 5](#_SPH_References). All references will be found in this list.

The code name SPHC is derived from “Smooth Particle Hydrodynamics” plus the computer language used = “C++”. It was developed in 1987 at Mission Research Corporation (now a division of ATK), in Albuquerque, NM as part of a program supported by the Defense Nuclear Agency (now part of the Defense Threat Reduction Agency) to model and understand high altitude shock physics. Subsequently, in an expanded version including the stress tensor, solid materials, and high explosives, it was used at Los Alamos National Laboratory and became the template for the LANL SPH code SPHINX. In 2003 it was modified to model aerospace engineering applications for NASA. At present (2020) it is owned and maintained by Stellingwerf Consulting. The code version discussed here is 1264.44 (code version 12.44, 64 bit compile). Setup files are designed to function independently of version number. Running these test cases on an earlier version of the code should work fine, except for a few new commands, which will be ignored. Running on a later version should always work.

The SPHC code is available via license agreement. An open source version of the code is available at [www.stellingwerf.com](http://www.stellingwerf.com). This version includes all code features, but omits strength of materials and explosive modeling modules. It is fully operational for fluid flow, mixing, and stability problems.

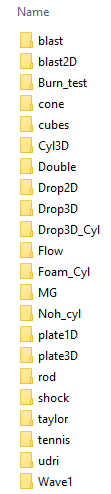
## Test Implementation

SPHC runs are usually stored in folders in /home//sdat (\home\sdat on Windows machines), but can be located elsewhere, as required.

The 22 tests in the current validation list are shown below. These folders are usually stored in the location

/home/sdat/test\_output on a Unix machine, or the equivalent location,

\home\sdat\test\_output on a Windows machine.

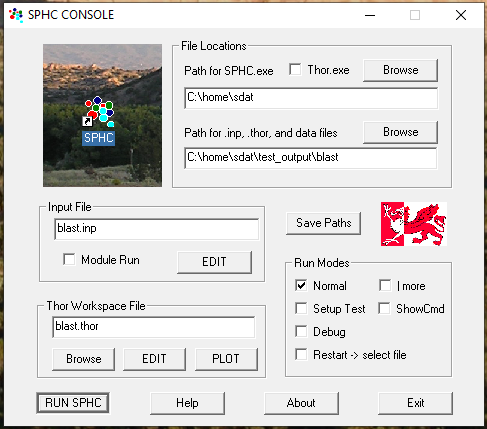


Each of these folders will contain an input (setup) file for the run (e.g. blast.inp), one or more plot files (e.g. blast.thor, history.thor), all of the plot files for the test run (e.g. p.00, p.01,…), all of the restart files for the run (e.g. s.00, .s.01, …), and the screen image summary (status\_final.txt), which shows the code version, run statistics, with the run time, resource usage, and other information at the endof the run

Variations on this file setup might be a folder named “/Test\_Output\_Fine”, which contains the identical runs, but run at higher resolution to determine the convergence behavior of the code, or “/test\_output\_small”, which are the same runs as the standard case, but including only the first and last plot dumps to limit the /test\_output file size. If this version is delivered, copy to a new folder called /test/output and run the test suite to fill out the other files (see below).

Any of the test cases can be run using the normal SPHC code launch methods. In UNIX installations, the command-line execution of the code is discussed in the SPHC User Manual. Batch execution consists of creating a file with a list of the appropriate “sphc” commands for each case.

In Windows installations, a user interface, SPHCInt is available to run individual cases. The interface is shown below (set up for the first test case), and discussed in detail in the SPHC User Guide.



For testing and validation (of a new installation, say), all of the tests are normally rerun, and the output is then compared to the standard set of runs or to the images shown in this document. Since machines and compilers differ in word length, etc., exact duplication of each run is not expected, but the results should reproduce all of the physical aspects of the test – such as shock strength and location, pressures, and other verifiable aspects of the test. The acceptable variations might include slightly different particle locations and distribution, local “jitter” in unstable regions, and other minor items. Many tests include geometry variations and symmetry assumptions, and the effects of these can be ascertained by rerunning with different conditions specified.

For Windows installations, a script is available to run all of the tests in simultaneously. All tests will usually run in this mode in under 5 minutes on a desktop workstation. The script name is TestDriver2.s, shown below. This script is written in the “S\_Tran” format, and requires the routine “stran.exe” to be assigned to the “\*.s” extension. This is the same scripting language that is used in SPHC setup files, in which case an onboard SPHC version is used that includes code setup extensions. See the “S-Tran User Guide” for coding details. The script format is mainly pseudo-code, and is fairly obvious in structure. Several formalities include: variable names starting with lower case letters are system commands and variables, those starting with upper case letters are user defined variables (defined when first used), and those starting with “$” are string variables. Three types of comments are allowed, as shown below. The test driver is listed below, note that some lines describing file locations, etc., may have to be modified for each installation. Also, the variables First\_run and Last\_run can be modified to select a subset of cases for testing.

**TestDriver2.s**

#---this version is set up for simultaneous running. Takes 5 min---

show\_line "-----SPHC test battery script2-----"

/\*---set up the directories for batch running:---------

\do\_runs (pick your name)

\run1 (pick your name)

run1.inp (same inp name)

\run2

run2.inp (same inp name)

...

Driver.s (this file)

-----------------------------------------------------\*/

#---put all strings with backslashes in quotes!!!---

$Exe\_folder = "c:\home\sdat\" // location of executable

$Run\_folder = "" // data folder. blank=local

First\_run = 1 // first case to do

Last\_run = 22 // last case to do

#---prompt for changes---

show\_nl

input $Exe\_folder //$Run\_folder

show\_nl

#---run list---

# directory name MUST MATCH the .inp file name!!!!

$Run[1] = shock; $Run[2] = cone; $Run[3] = plate1d

$Run[4] = udri; $Run[5] = rod; $Run[6] = tennis

$Run[7] = cubes; $Run[8] = taylor; $Run[9] = plate3d

$Run[10] = mg; $Run[11] = double; $Run[12] = foam\_cyl

$Run[13] = noh\_cyl; $Run[14] = drop2D; $Run[15] = drop3D

$Run[16] = wave1; $Run[17] = burn\_test;$Run[18] = cyl3D

$Run[19] = blast; $Run[20] = blast2D; $Run[21] = flow

$Run[22] = drop3D\_cyl

#---execute the runs in a loop---

for\_i = First\_run Last\_run

$run\_folder = $Run\_folder $Run[i] "\"

#---first delete former results---

$str = "del " $run\_folder "\s.\*" // use appropriate system command

system $str

#---now do the run---

$str = "start " $Exe\_folder

$str = $str "sphc i " $Run[i] ".inp " $run\_folder " >" $run\_folder "\screen.txt"

show\_nl

show\_field $date ": "

show\_field "Beginning run " $Run[i] ....

system $str // call SPHC here

end\_i

show\_nl

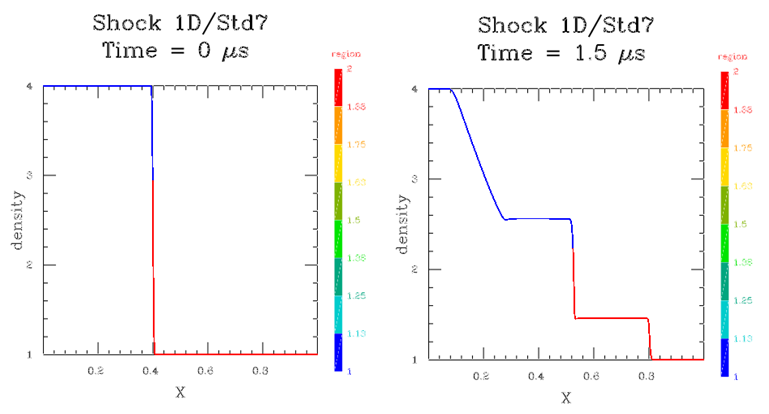
Verification of each result is most easily done by execution of the “*case*.thor” file in each run folder, which usually is set to show the setup configuration, the stepping through the plot dumps until the final result is seen. Here we show the initial and final results that are considered to be “correct”, as well as a few other plots that can be checked if needed.

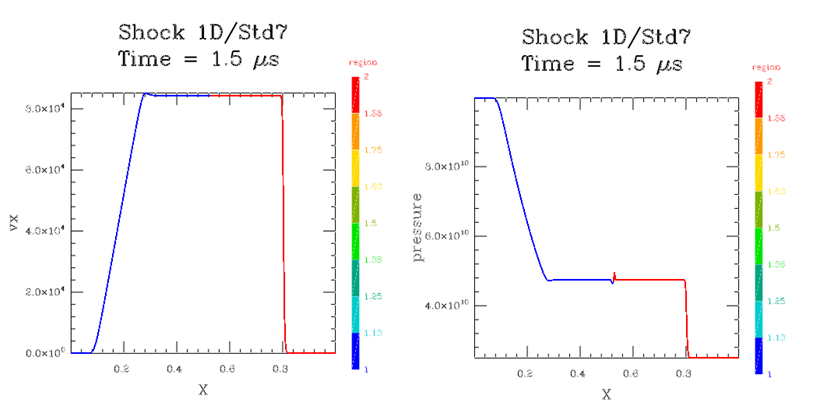
## Test Descriptions and Results

### Shock Tube – 1D

Computing a simple shock tube solution was the first major hurdle for the SPH technique. Unlike a grid based code, the SPH particles for a fluid are unconstrained, and unless carefully controlled, they easily overrun and penetrate adjacent particles near the shock front even in the weak shock case. It took most of the 1975-85 decade to perfect the specialized approach to this problem used in SPHC. See Monaghan and Gingold 1983 for a description of the basic SPH techniques required for shocks. Since these techniques are so specific and so sensitive, we have found that even a small deviation from the following accurate test case should be regarded as a serious code problem that needs fixing. These methods are now built into the code defaults and require no direct user seettings for most problems.

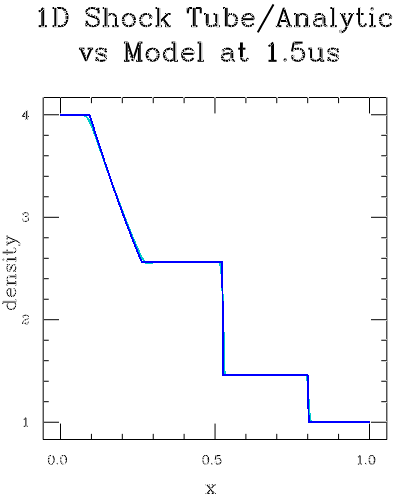
This test is a 1 dimensional (planar) shock tube test case. We model a tube of length 1 cm (0-1) filled with ideal gas, and having a membrane at location 0.4. Density to the left of the membrane is 4 g/cc, to the right is 1 g/cc. Scaling to more reasonable densities does not affect the results. Both sides have temperatures of 300K, mean molecular weight of 1, and gas gamma of 5/3. At time 0 the membrane is burst, and the development is followed for 1.5e-6 sec, at which time the shock has propagated to 0.80 cm, the contact surface is a density jump at about 0.50, and a rarefaction wave has propagated to the left. Run time for this case was 2.84 s.



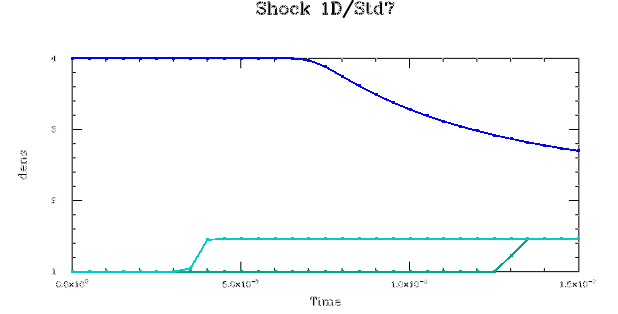


Velocity and pressure variations are shown above. The pressure in the SPHC run has a tendency to spike at the interface, caused by the abrupt jump in the initial condition, which cannot be modeled by a smoothing code. The acceptably small effect shown here is due to a setup smoothing operation (last line of setup, below). This should be included in all problems with density discontinuities at time 0.

This case has an analytic solution, easily obtained from any fluid dynamics text. The following figure shows the comparison of the model result (light blue) to the analytical result (dark blue). The model shows some slight smoothing, but matches the desired result very closely. This agreement improves with finer zoning (more particles), as expected.



Data probes are defined at X locations 0.25, 0.50, and 0.75. The “history.thor” file produces the following history plot for the density variation at these locations.



The full listing for this test case is shown below ( file = shock.inp).

#====shock tube test case====

problem\_title "Shock 1D"

run\_title "Std7"

dimension = 1

nparticles = 500

max\_time = 1.5e-6

restart\_dumps = 15

hist\_dumps = 30

dump\_accel

dump\_eos

#---------boundaries-------

set\_boundary left

location = 0.

side low

end\_boundary

set\_boundary right

location = 1.

side high

end\_boundary

#-------data probes--------

set\_probe fixed 0.25 0. 0.

set\_probe moving 0.5

set\_probe fixed 0.75

#-------regions----------

set\_region "high den"

material pg

eos pg

gamma = 1.6667

mu = 1.

density = 4.

temp = 300.

end\_region

set\_region "low den"

material pg

eos pg

gamma = 1.6667

mu = 1.

density = 1.

temp = 300.

end\_region

#---- model build follows----

begin\_region "high den"

part\_mult = 0.4

do\_block 0.4 1.0 1.0

translate\_reg 0.2 0. 0.

begin\_region "low den"

part\_mult = 0.6

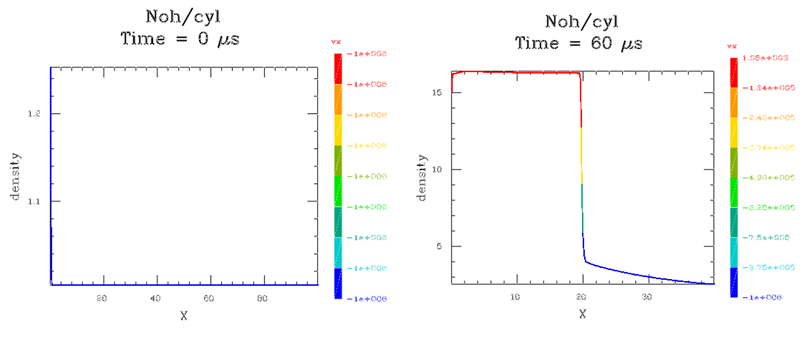
do\_block 0.6 1.0 1.0

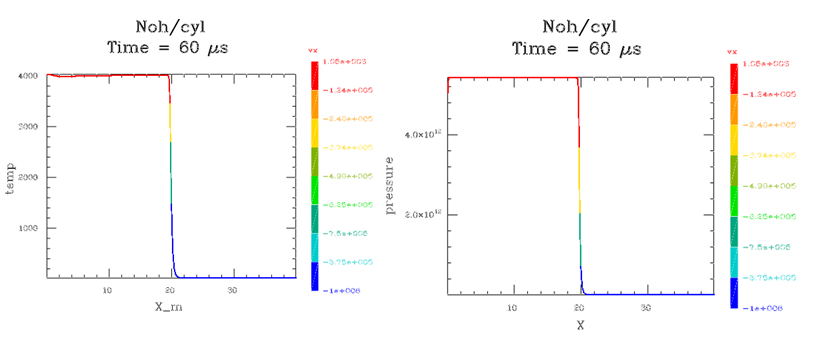
translate\_reg 0.7 0. 0.

smooth 2 1 1

### Noh Shock – 1D

The “Noh” test is a variation on the shock tube in which a single region is used, moving toward a wall or origin at high velocity (see Noh, 1978). The SPHC test case is run in cylindrical geometry (flow converges toward an axis) at 1.e6 cm/s, or about Mach 30 for the ideal gas with gamma = 5/3. This a standard test for problems involving extreme compression, and has an analytic, self-similar solution. Run time for this case was 2.14 s.





Most codes produce excess heating on axis for this problem. Only a hint of this is seen here in the form of a slight density dip on axis. In SPHC special treatment is needed to avoid this and other axis issues. The solution used here is a small offset from the axis at time 0. This also works for the planar and spherical cases.

The full listing for this test case is shown below ( file = noh.inp).

#====shock tube test case====

problem\_title "Noh"

/\* Noh problem, cyl or sphere case \*/

/\* Result is very sensitive to initial setup near origin \*/

#---units---

Usec = 1.e-6

Mtr = 100

dimension = 1

nparticles = 400 // 100 runs, but rough 400 ok 800 better

Sphere = false // spherical case, false for cyl

if Sphere run\_title = "sph"

if ~Sphere run\_title = "cyl"

if Sphere spherical

if ~Sphere cylindrical

max\_time = 60\*Usec

restart\_dumps = 6

hist\_dumps = 60

pert\_size = 0.

h\_inp = 1.5

h\_vary = true

quiet\_start\_solid false

debug\_part = 2

energy\_smooth .2 0

#---viscous diffusion---

// needed in spherical case

if Sphere

av\_g1 = 1

av\_g2 = 0

end\_if

dump\_eos

Vel = -1.e6 // inward velocity

Shift = .01 // exclude singular origin

#---------boundaries-------

set\_boundary left

location = Shift

side low

end\_boundary

set\_boundary right

location = 1\*Mtr+Shift

side high

velocity Vel

end\_boundary

#-------regions----------

set\_region liner

material pg

eos pg

mu 1

gamma 5/3

density = 1

temp = 10 // should be 0, code can't do 0

end\_region

#---- model build follows----

set\_no\_neg 1 0 0 // needed for planar setup

begin\_region liner

part\_mult = 1

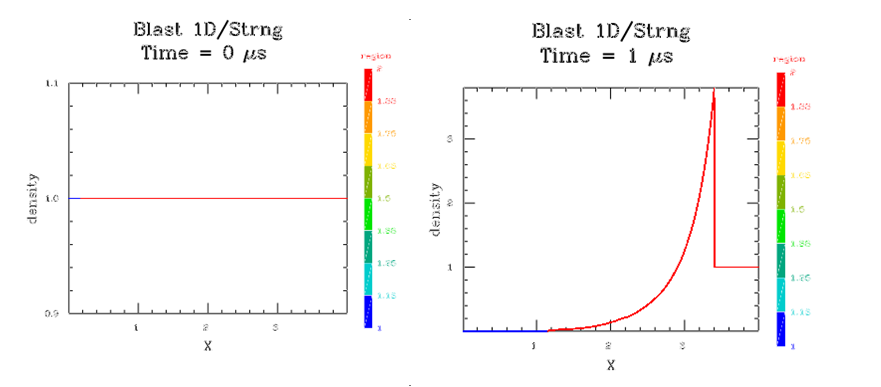
do\_block 2\*Mtr 0 0

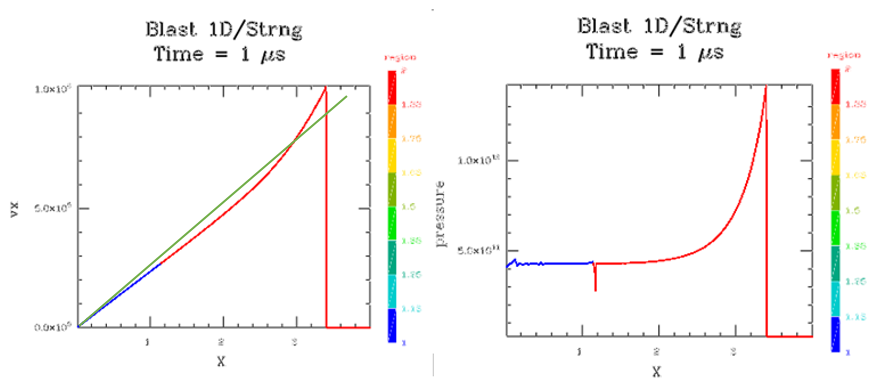
translate\_reg Shift 0 0

velocity\_reg Vel 0 0

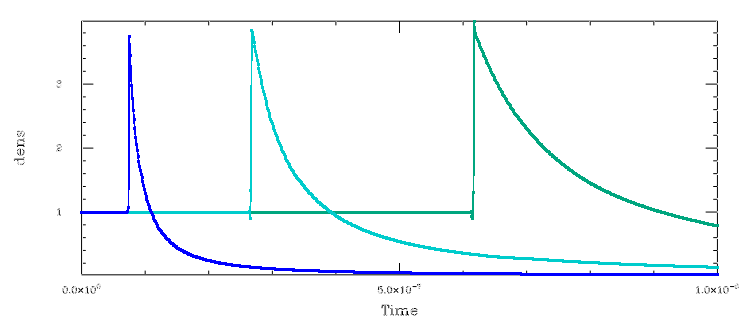
### Blast Wave - 1D

This test case is often referred to as the “Sedov” similarity solution, but was first derived by von Neumann and Taylor in 1941 – see the Los Alamos report LA2000 (Bethe, et al 1947) for details. This test models the release of a large amount of energy at a point in space resulting in a strong, spherical shock front propagating outward. The properties of the atmosphere (i.e.  of an ideal gas) are the only parameters. This case models a 1 dimensional spherical explosion, the next case repeats the identical test, but using a 2 dimensional model. The green line shown in the final velocity plot shows the theoretical slope for this case. The deviation is caused by the finite sized central initial core. Run time for this case was 18.40 s.





Three data probes are defined at 0.30, 0.50, and 0.70 in X. The “history.thor” plot produces the plot below. The larger time gap between the two rightmost peaks indicate a slowing of the front.



The full listing for this test case is shown below ( file = blast.inp).

#====blast wave test case====

problem\_title "Blast 1D"

run\_title "Strng"

dimension = 1

nparticles = 1000

max\_time = 1.0e-6

restart\_dumps = 20

plot\_dumps 20

plot\_press

hist\_dumps = 800

err\_tol 0.01

dump\_accel

dump\_eos

spherical // spherical wave!

Outer = 4

Bdry = 0.04\*Outer

#---------boundaries-------

set\_boundary left

location = 0.

side low

end\_boundary

set\_boundary right

location = Outer

side high

end\_boundary

#-------data probes--------

set\_probe fixed 0.3\*Outer 0. 0.

set\_probe fixed 0.5\*Outer 0 0

set\_probe fixed 0.7\*Outer 0 0

#-------regions----------

set\_region "high temp"

material pg

eos pg

gamma = 1.6667

mu = 1.

density = 1.

temp = 1e8

end\_region

set\_region "low temp"

material pg

eos pg

gamma = 1.6667

mu = 1.

density = 1.

temp = 300.

end\_region

#---- model build follows----

begin\_region "high temp"

part\_mult = Bdry/Outer

do\_sphere Bdry

begin\_region "low temp"

part\_mult = (Outer-Bdry)/Outer

do\_sphere Outer Outer-Bdry

smooth 2. 1 1

### Blast Wave – 2D

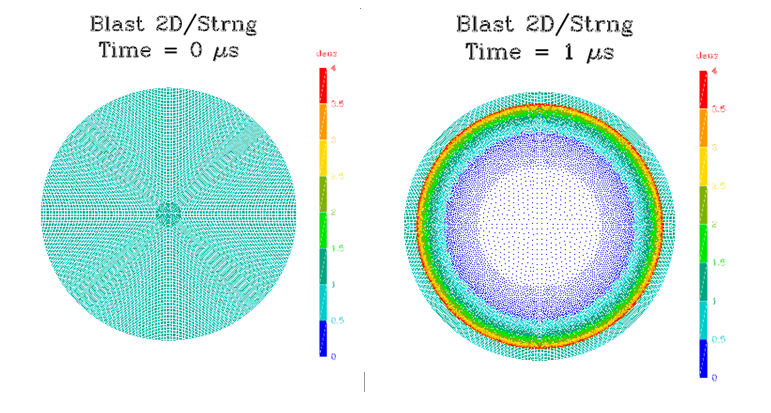
This test case illustrates the type of changes needed to run a 2D version of the blast wave case shown above. The setup here is to define a ¼ sphere in the X/Y plane with the “cylindrical” option (Y axis) and “symmetry y” to install a reflect boundary at Y = 0. This will result in a full sphere simulation the same as the 1D case with “spherical” specified, above.

In many SPHC runs simply changing the dimension variable from 1 to 2 or 3 and adding extra particles will convert to the dimension desired. In this case several other changes were required.  
1. Probe locations are moved slightly off axis to avoid axis effects.

2. The radius of the hot inner region was extended from 0.04 to 0.10 in order to include a few more hot particles at the axis at time 0. Another way to do this would be to use a “grid\_ratio” command.

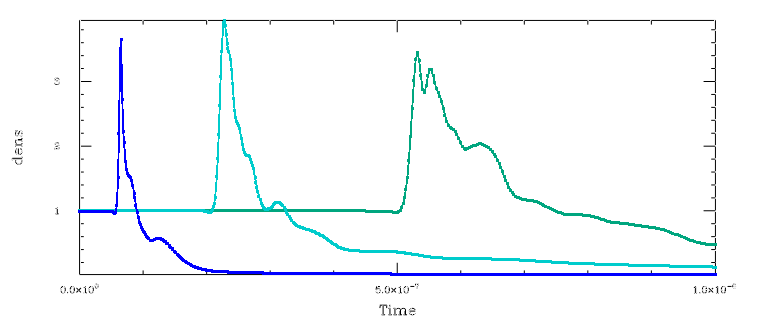
3. The temperature of the extended hot region was lowered from 1.e8 to 8.e6 to make the energy of the inner region approximately the same as the 1D test case. This correction is not exact and will result in some small deviations.

Run time for this case was 38 s.



The perfect curcular shape of the blast front is a test of the “cylindrical’ code option. The very small perturbations along the horizontal and vertical axes show that the symmetry treatment in X and Y, and at the origin is working. The roughness along the contours of different density colors (esp. light and dark blue) are caused by the early stage of a physical instability between the hot and cool regions of the blast flow that develops at later time. See the “Explosion” section, below, for more details of this instability.

Comparing the history plot to the 1D case, the results are similar in amplitude, timing and shape, but with some additional roughness due to the 2D zoning that the fixed probes cannot entirely resolve (the probe routine sums over nearby particles at each time, and some variations are expected). In applications, use more particles to overcome this.



The full listing for this test case is shown below (file = blast2D.inp).

#====blast wave test case====

problem\_title "Blast 2D"

run\_title "Strng"

dimension = 2

nparticles = 10000

max\_time = 1.0e-6

restart\_dumps = 20

plot\_dumps = 40

hist\_dumps = 600

err\_tol 0.01

dump\_accel

dump\_eos

cylindrical // makes a spherical wave

symmetry y

Outer = 4

Bdry = 0.1\*Outer // increased to help resolution

#---------boundaries-------

set\_boundary outer

location = Outer

side high

direction r

end\_boundary

#-------data probes--------

set\_probe fixed 0.3\*Outer 0.1 0.

set\_probe fixed 0.5\*Outer 0.1 0

set\_probe fixed 0.7\*Outer 0.1 0

#-------regions----------

set\_region "high temp"

material pg

eos pg

gamma = 1.6667

mu = 1.

density = 1.

temp = 8e6 // to match the 1D case energy

end\_region

set\_region "low temp"

material pg

eos pg

gamma = 1.6667

mu = 1.

density = 1.

temp = 300.

end\_region

#---- model build follows----

begin\_region "high temp"

part\_mult = (Bdry/Outer)^2

do\_sphere Bdry

begin\_region "low temp"

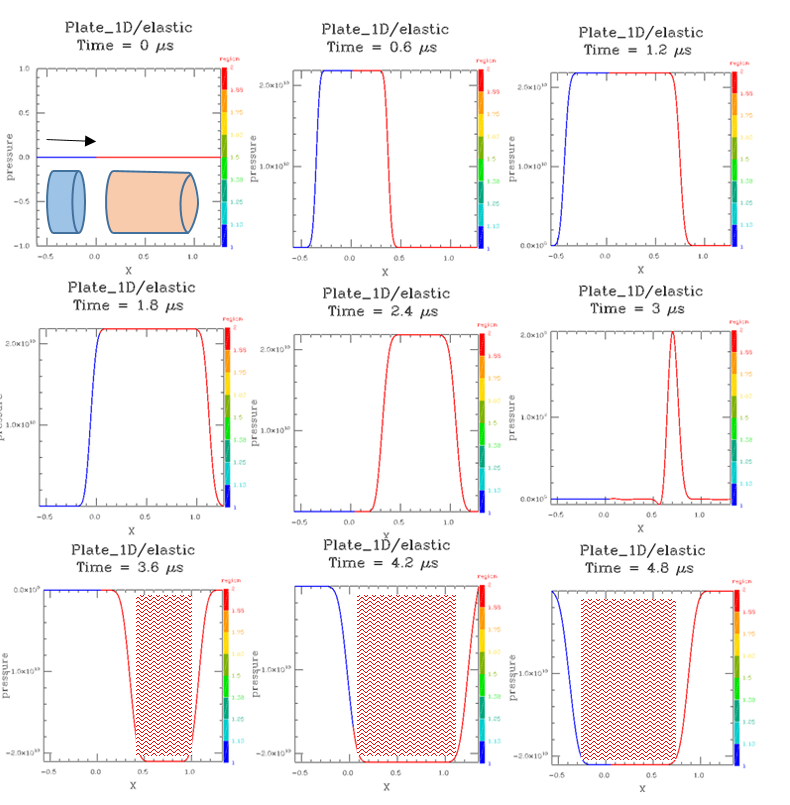
part\_mult = ((Outer-Bdry)/Outer)^2

do\_sphere Outer Outer-Bdry

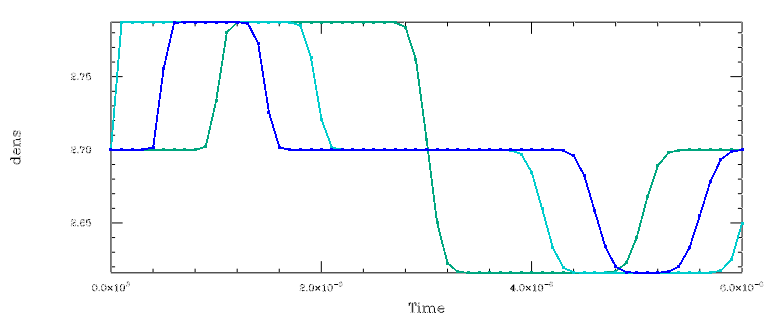
smooth 2. 1 1

### Flyer Plate – 1D

This is a standard test of an impact of an aluminum plate travelling at about the speed of sound in air (0.39 km/s, or 1181 ft/s) with another aluminum plate with twice the thickness of the travelling plate. The physics for this test is simplified (linear elastic equation of state). The main result of the test is the generation of strong shocks which propagate to the ends of the stationary plate, reflect, and then converge to form a strong rarefaction wave exactly at the center of the second plate. Early forms of SPH codes showed strong numerical instability at this point (shaded area, below. No pressure fluctuations are seen in this test. The present code is designed to avoid this instability, and this test is a test of this. Run time for this test is 28 s.



Data probes are included in this run at the interface and in the interior of the plates on either side. The density variation is shown below.



The full listing for this test case is shown below (file = plate1d.inp).

#====flyer plate test case====

problem\_title "Plate\_1D"

run\_title "elastic"

dimension = 1

nparticles = 500

//space\_adjust = 1.02

max\_time = 6.e-6

restart\_dumps = 30

hist\_dumps = 60

pert\_size = 0. // no rand

pmin = -1.e11 // no pmin

dump\_accel

dump\_eos

#-----user variables-----

Veloc = 0.39e5

Thick1 = 0.608 // left slab

Thick2 = 1.27 // right slab

#-------strength-------

strength\_model elastic

#-------regions----------

set\_region "moving"

material al

//eos usup

eos linear

end\_region

set\_region "fixed"

material al

//eos usup

eos linear

end\_region

#-------data probes--------

set\_probe moving -Thick1/2

set\_probe moving 0.

set\_probe moving Thick2/2

#---- model build follows----

begin\_region "moving"

part\_mult = Thick1/(Thick1+Thick2)

do\_block Thick1 1.0 1.0

translate\_reg -Thick1/2 0. 0.

velocity\_reg Veloc 0 0

begin\_region "fixed"

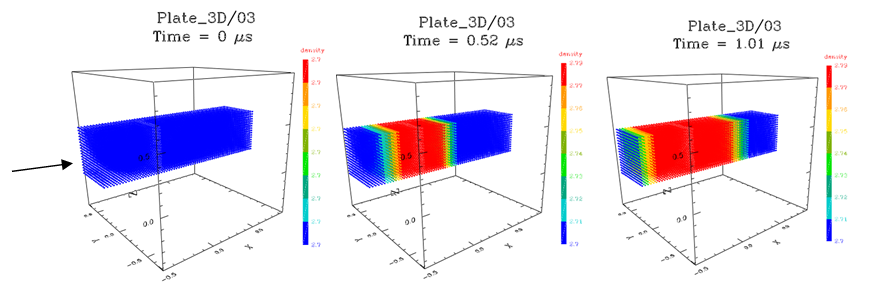
part\_mult = Thick2/(Thick1+Thick2)

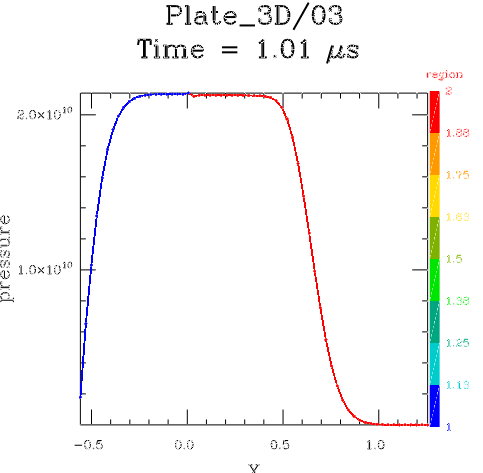
do\_block Thick2 1.0 1.0

translate\_reg Thick2/2 0. 0.

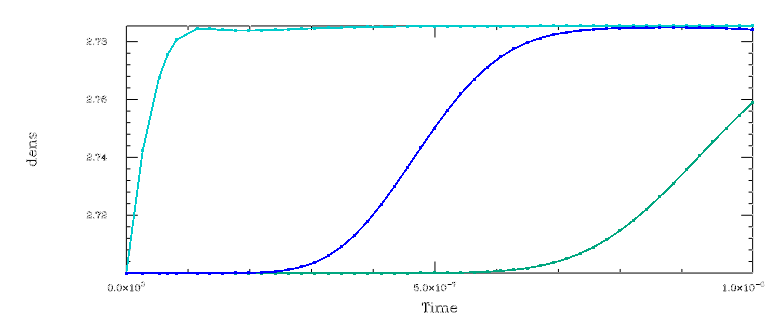
### Flyer Plate – 3D

This is the same case discussed above, but for the 3D case. Movement is in the X direction, symmetry conditions are used in th Y and Z directions, and “reflect” boundaries are used in front and on the top sides. The side view pressure plot compares well with the 1D results shown above. To save time this test is only run to 1 s, but could be extended if needed. Run time for this case is 84 s.





Probe data for this case is shown below.



Full listing for this case is shown below – file is plate3d.inp.

#====3D flyer plate test case====

problem\_title "Plate\_3D"

run\_title "03"

dimension = 3

nparticles = 20000

space\_adjust = 1.05

max\_time = 1.e-6

restart\_dumps = 10

hist\_dumps = 50

pert\_size = 0. // no rand

h\_inp = 1.

pmin = -1.e11 // no pmin

dump\_accel

dump\_eos

#-----user variables-----

Veloc = 0.39e5

Thick1 = 0.608 // left slab

Thick2 = 1.27 // right slab

#-------strength-------

strength\_model elastic //off for inst

#-------regions----------

set\_region "moving"

material al

//eos usup

eos linear

end\_region

set\_region "fixed"

material al

//eos usup

eos linear

end\_region

#-------data probes--------

set\_probe moving -.3

set\_probe moving 0.

set\_probe moving 0.6

#-------boundaries--------

symmetry y

symmetry z

set\_boundary top

location 0.5

direction y

side high

type reflect

end\_boundary

set\_boundary front

location 0.5

direction z

side high

type reflect

end\_boundary

#---- model build follows----

begin\_region "moving"

part\_mult = Thick1/(Thick1+Thick2)

do\_block Thick1 1.0 1.0

translate\_reg -Thick1/2 0. 0.

velocity\_reg Veloc 0 0

begin\_region "fixed"

part\_mult = Thick2/(Thick1+Thick2)

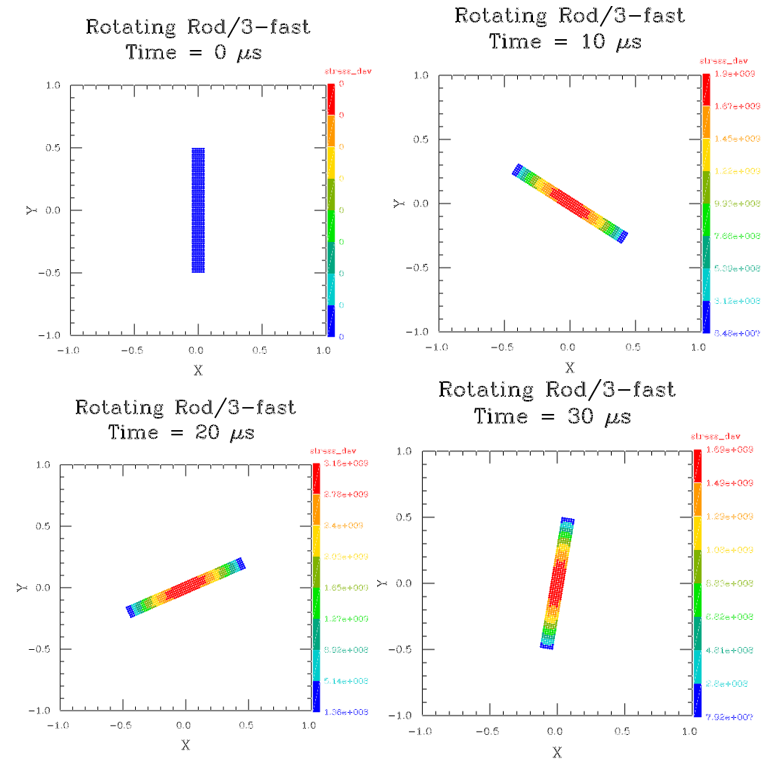
do\_block Thick2 1.0 1.0

translate\_reg Thick2/2 0. 0.

### Rotating Rod – 2D

This is a test of the conservation of angular momentum in the new Virtual Stress Point (VSP) feature of the SPH implementation in SPHC. “Classical SPH” (see Libersky, et al 1993) cannot handle large angle rotation of objects due to stress tensor edge errors for solid objects. VSP uses a centering technique on the stress tensor terms to eliminate these errors and allow arbitrarily large rotations.

Although the test is called “rotating rod”, and a do\_cylinder command is used in the setup deck, in order to facilitate a rapidly running test case, this version is run in 2D, in which case the same setup produces a model of a long plate seen edge-on rotating around its long axis. Since the plate starts with zero stress, the rotation causes some initial oscillations in length and stress field of the rod, Maximum deviatoric stress is at the center of the plate, and settles down to about 2.e9 cgs ( 2 kBar) at 100 micro-s. Stress at the ends of the plate cross section remain zero, as expected. The rotational velocity of the plate remains exactly constant for any run length. This feature of the code is especially important for high energy simulations producing rapidly rotation debris.



The setup file for this case is found in “rod.inp”. This case runs in 16 s.

#==== rotating rod test case ====

problem\_title = "Rotating Rod"

run\_title = 3-fast

dimension = 2

nparticles = 500

strength\_model elastic

max\_time 1.e-4

dump\_accel

space\_adjust = 1.1

pert\_size = 0

error\_control false

pmin = -1.e11

restart\_dumps = 20

hist\_dumps = 100

#----region def----

set\_region rod

material al6061

eos usup

track\_com

end\_region

#--------model build------

begin\_region rod

part\_mult 1

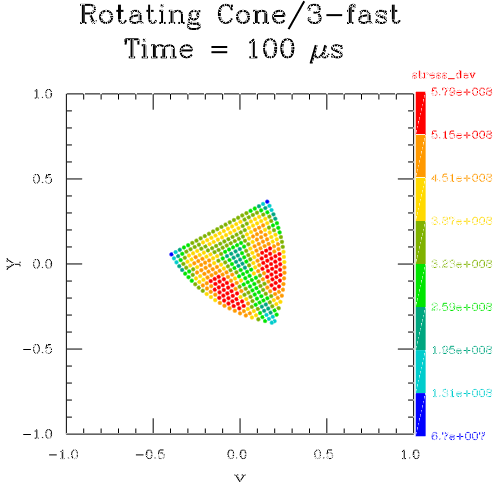
do\_cylinder .05 1

spin\_reg 0 0 1.e5

### Rotating Cone – 2D

This is a test of the “do\_arc\_cone” command, which is usually used to construct a simple nose cone. It also tests the rotational properties of the code, as in the previous test case.

In this case we set up a long rod with an arc cone cross section, then rotate the rod about its center line rapidly to study the resulting stress field in this non-symmetrical shape. Surprisingly, the maximum stress in the cone are not found at the center, as in the rotating rod test, but symmetrically off-center.

The run setup is found in file cone.inp. This test runs in 10 s.

#==== rotating cone test case ====

problem\_title = "Rotating Cone"

run\_title = 3-fast

dimension = 2

nparticles = 500

strength\_model elastic

max\_time 1.e-4

dump\_accel

space\_adjust = 1.1

pert\_size = 0

restart\_dumps = 20

hist\_dumps = 100

#----region def----

set\_region rod

material al6061

eos usup

track\_com

end\_region

#--------model build------

begin\_region rod

part\_mult 1

do\_arc\_cone .33 .66

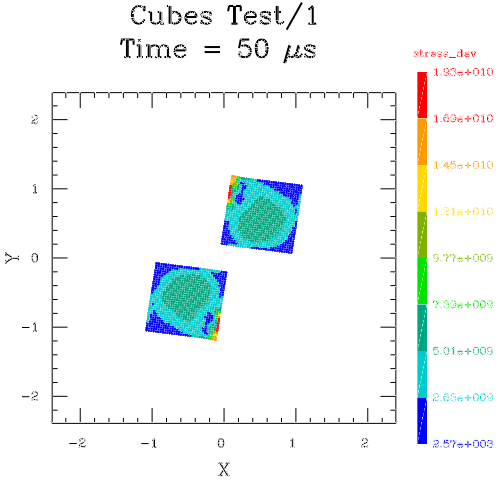
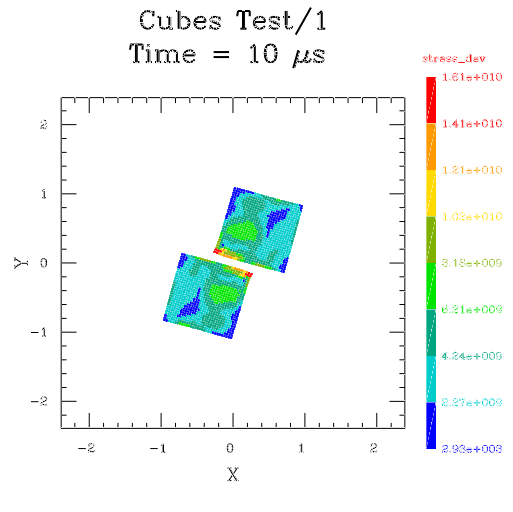
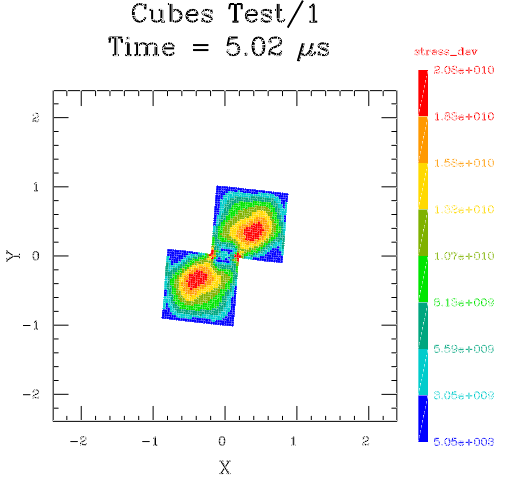
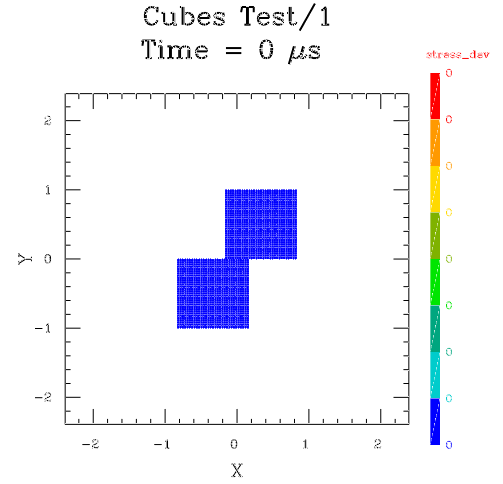
center\_com // fine tune centering

spin\_reg 0 0 1.e5

### Cube Impact – 2D

This test models the impact at relatively low velocity (2,000 cm/s = 0.20 km/s) of two off-center aluminum cubes (modeled in 2D as long rods) and the subsequent elastic rebound in free space. The resulting motion is a combination of elastic oscillations and rotation. This test is to see if the rebound is reasonable and stable, and no analytic or experimental comparisons are used. Adding comparison data could be done, but would then become a test of the material model, which is not intended in this case.

These plots are colored on deviatoric stress.



The input file for this case is “cubes.inp”. Run time is 23.45 sec to 50 sec model time..

#====colliding cubes====

problem\_title "Cubes Test"

run\_title "1"

C\_size = 1

C\_vel = 2.e4

dimension = 2

nparticles = 2000

space\_adjust = 1.05

max\_time = 5.e-5

restart\_dumps = 20

hist\_dumps = 80

error\_control false

dump\_eos

dump\_accel

#--------strength--------

strength\_model elastic

slip\_regions

#-------regions----------

set\_region cube1

material al1350

eos usup

end\_region

set\_region cube2

material al1350

eos usup

end\_region

#---- model build follows----

begin\_region cube1

part\_mult = 0.5

do\_block C\_size C\_size C\_size

velocity\_reg 0. -C\_vel 0.

translate\_reg C\_size/3 C\_size/2 0

begin\_region cube2

part\_mult = 0.5

do\_block C\_size C\_size C\_size

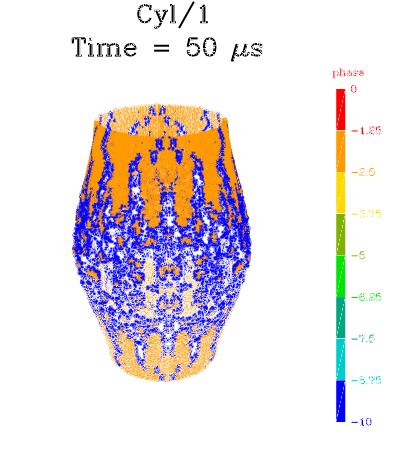
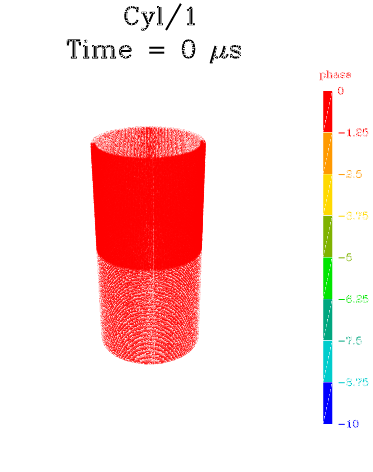
velocity\_reg 0. C\_vel 0.

translate\_reg -C\_size/3 -C\_size/2 0

### Cylinder Fracture – 3D

This is a simple case of a debris field created from an object completely fractured via expansion. This expansion is normally caused by an interior explosion. An explosive event imparts a roughly radial acceleration on surrounding structures, which continues until fracture allows the hot gas to vent. From this point on, the outward velocity of the fragments are constant until atmospheric drag forces cause deceleration. Here we simulate such an event by starting with an intact cylinder, but with a radially outward velocity field. This causes motion that induces fracture that depends on the material, strength model and fracture model specified. In this case the material is Aluminum 6061, the strength model is elastic-perfectly plastic, and the fracture model is the usual strain-to-fracture criterion, with a Weibull distribution of failure strengths between SPH particles. The initial velocity of the material is set to 40,000 cm/s (1312 ft/s) at the radius of the cylinder, but linearly decreases to 0 at 4 times the radius (i.e. near the ends). This simulates the decrease in blast effect farther away from the origin. This test is particularly useful for evaluating the effects of the various fracture parameters quickly and easily, before adding the details of any particular scenario.

These are the initial and final states of the run, colored on phase. Full symmetry is assumed, which allows the 7072 particles to nicely resolve 1/8 of the full cylinder. The reflected segments are filled in with the Thor plotter.



Further dissuasion of this type of simulation can be found in the “Explosion” section, below.

This is the input setup file “cyl3D.inp”. Run time for this test is 1.42 min.

#==== exploding egg test case====

problem\_title "Cyl"

run\_title "1"

Usec = 1.e-6

dimension = 3

nparticles = 10000

space\_adjust = 1.10

max\_time = 50\*Usec

restart\_dumps = 5

plot\_dumps = 20

hist\_dumps = 40

h\_inp = 1.0

h\_vary = true

error\_control false

pert\_size 20 // (up from 1)

symmetry x

symmetry y

symmetry z

#--------strength--------

strength\_model elas\_perf\_plas

fracture

#-------regions----------

set\_region ball

material al

eos grun

strength\_model elas\_perf\_plas

fracture

weibull

end\_region

#---- model build follows----

Vel = 4.e4

Radius = 3

Thick = 0.20

begin\_region ball

part\_mult = 1

do\_cylinder Radius 4\*Radius Thick

radial\_velocity

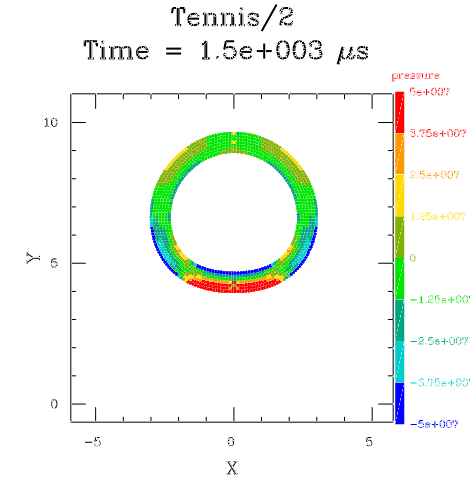
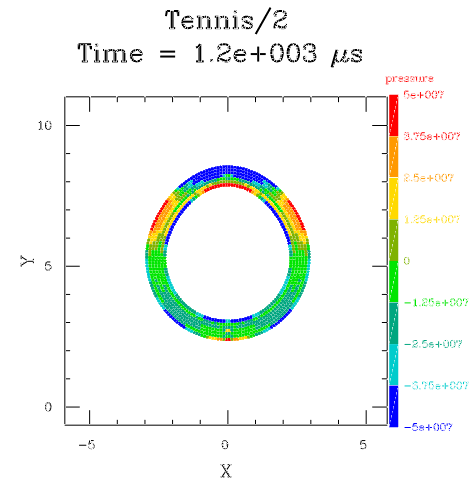
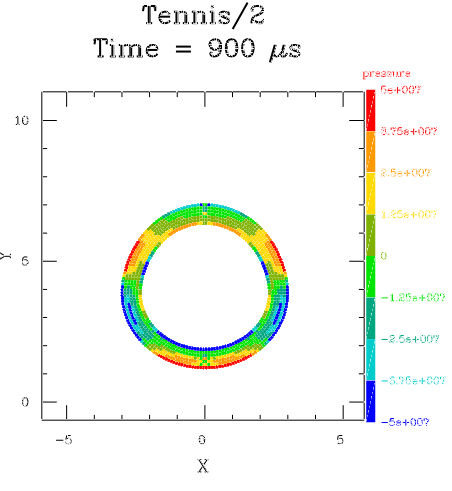
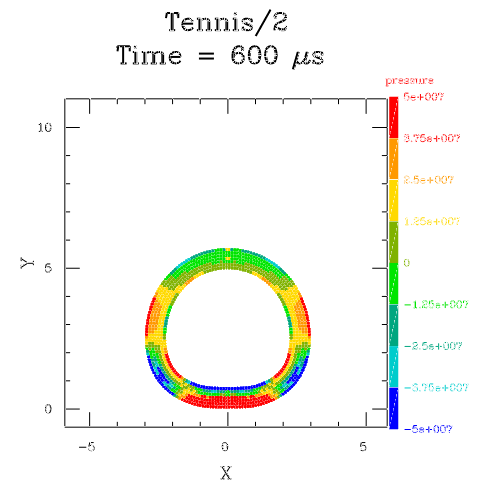
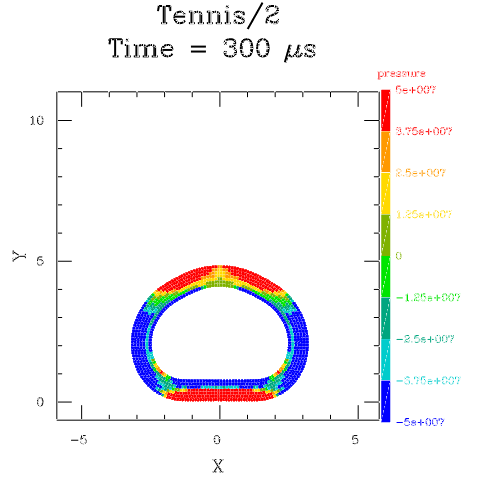
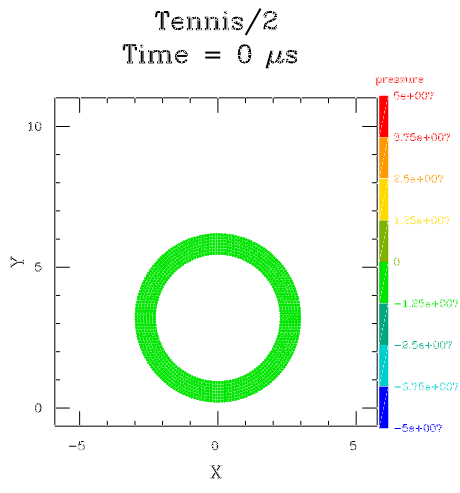
points 0 4\*Vel Radius Vel 4\*Radius 0

end\_velocity

### Tennis Ball – 2D

This is a classic simulation of a hollow elastic tennis ball impacting a wall at 112 mph (5000 cm/s = 164 ft/s). At this speed the sphere shows significant shape distortion, and large tensile forces develop. In some early SPH codes, this is a prime scenario for the “tensile instability” to develop, causing the ball to fracture like a Christmas-tree ornament. The SPH run is perfectly stable.

These plots are colored on volumetric pressure, which can cause instability when negative.



Setup shown below. Run time is 16.5 s.

problem\_title "Tennis"

run\_title "2"

dimension = 2

nparticles = 600

space\_adjust = 1.03

max\_time = 20.e-4

restart\_dumps = 20

hist\_dumps = 40

h\_inp = 1.0

h\_vary = true

error\_control false

#-------regions----------

set\_region ball

material rub

eos grun

pmin = -1.e11

end\_region

#--------strength--------

strength\_model elastic

#------boundary-----

set\_boundary wall

direction y

slip

end\_boundary

#--this gives cylinder

//symmetry x

#--or...this gives a sphere

cylindrical

#---- model build follows----

begin\_region ball

part\_mult = 1

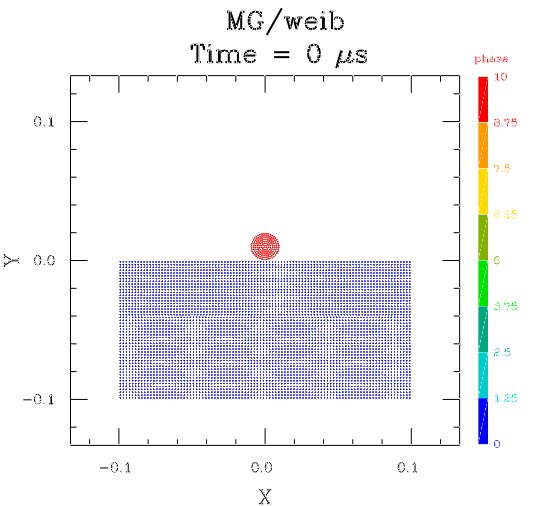
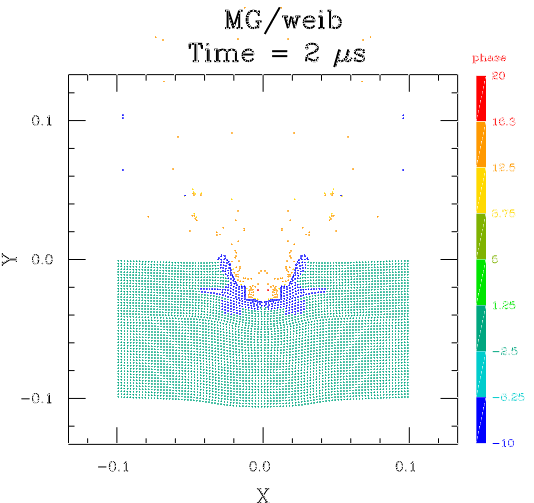
do\_sphere 3.0 .75

translate\_reg 0. 3.2 0.

velocity\_reg 0. -5.e3 0. // 100 mph

### Solar Panel Impact – 2D

This is a simplified model of a spherical water drop impacting a space station solar array (layers of glass and kapton – a new material defined in this setup) at 112 mph (5000 cm/s = 164 ft/s). Such liquid is sometimes ejected from docked vehicles, and can cause damage if not properly configured. These results have been compared to experimental damage, as well as observed damage on the solar arrays. This run uses a “Weibull” fracture model for the glass layers. Color in these plot represents “phase”, defined as -10 = ‘fractured”, 0 = “solid”, 10 = “liquid”, 20 = “vapor”.

Setup shown below. Run time is 16.5 s.

A feature of this setup is that it retains the “Sphinx” input format, which requires all input quantities to be numeric values. An updated SPHC input would ordinarily assign local variable names to all input quantities and show computed values as algebraic formulae, allowing variations to be easily specified. Note that SPHC will run SPHINX input files, with only minor adjustments.

#====droplet on solar panel====

# Maria Greene's setup (Boeing) of a water drop

# hitting a solar panel on station

# Water comes from a docked shuttle

problem\_title "MG"

run\_title "weib"

dimension = 2

nparticles = 2500

cylindrical

space\_adjust = 1.05

max\_time = 2.e-6

restart\_dumps = 20

hist\_dumps = 20

err\_tol 0.01

slip\_regions 0.5

#------boundaries--------

set\_boundary outer

location 0.1

side high

type fixed

end\_boundary

#------materials---------

add\_material kapton

rho\_0 = 1.56

cs\_0 = 5.376e5

cv\_0 = 1.09e7

s\_shock = 1.55

gamma\_G = 2.10

ey = 2.6e10

pr = 0.34

sy = 0.69e9

st = 1.72e9

em = 0.75

end\_material

#--------strength--------

strength\_model elas\_str\_hard

fracture

#-------regions----------

set\_region sphere

material h2o

eos grun

end\_region

set\_region plate

material glass

eos grun

weibull .1 1

end\_region

set\_region scell

material glass

eos grun

weibull .1 3

end\_region

set\_region subst

material kapton

eos usup

end\_region

#---- model build follows----

# this is SPHINX style input, with numerical args

# for SPHC prefer to define variables instead

begin\_region sphere

part\_mult = 0.03

do\_sphere 0.01

velocity\_reg 0. -3.e5 0.

translate\_reg 0 .01 0

begin\_region plate

part\_mult = 0.2

do\_cylinder 0.1 0.02

translate\_reg 0.0 -0.01 0.

begin\_region scell

part\_mult = 0.2

do\_cylinder 0.1 0.02

translate\_reg 0.0 -0.03 0.

begin\_region subst

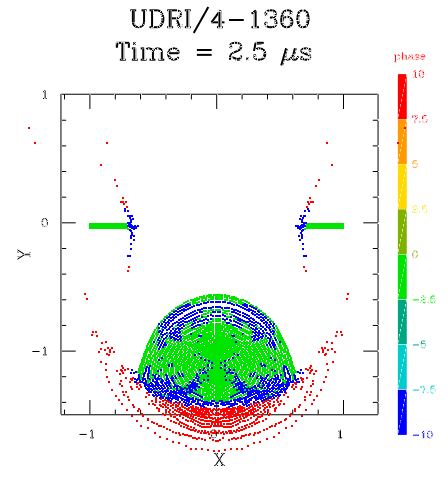
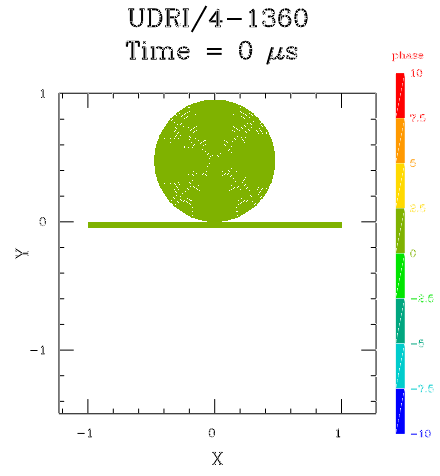
part\_mult = 0.6

do\_cylinder 0.1 0.06

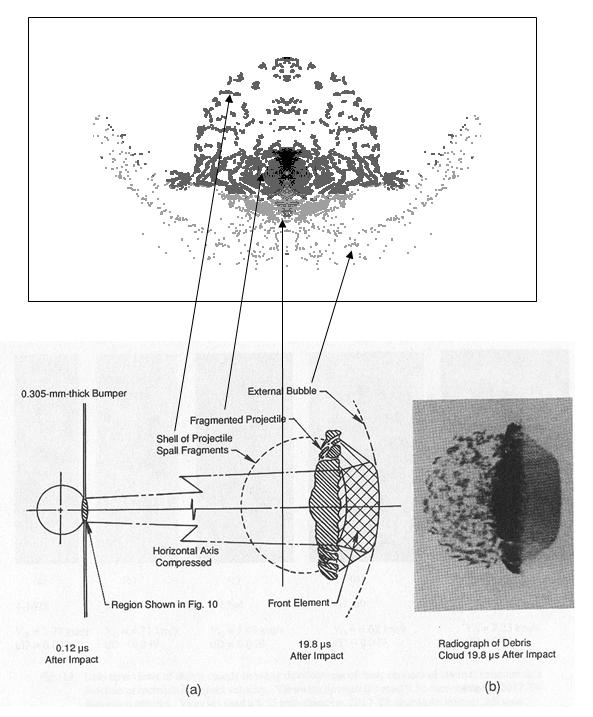
translate\_reg 0.0 -0.07 0.

### Ball on Plate Impact – 2D

This is a classic “Ball on Plate” impact in which a sphere (diameter = 4.765 mm) travelling at about orbital velocity (6.6 km/s) impacts a thin aluminum plate (Whipple shield). This case (labeled “udri”) models experiment number 1360 carried out at the University of Dayton. The plots show the initial and final configurations of the run colored on “phase” (blue = fractured, red = melted). Note that the lower region of the final plot shows melted material at the bottom, and spalling (fractured shell material) at the rear (top).



This run can be extended to 10 s to compare with a UDRI X-Ray photo of this experiment, shown below (model at the top, annotated photo at the bottom). The spalled material. As well as the liquid frontal volume are clearly visible in each case, and the agreement between model and experiment is excellent.



This run takes 5.4 s and is labeled “udri”. Listing below.

#====ball on plate test case====

problem\_title "UDRI"

run\_title "4-1360"

#----basic 2D settings---

dimension = 2

nparticles = 2000

space\_adjust = 1.05

cylindrical

max\_time = 2.5e-6

restart\_dumps = 25

hist\_dumps = 50

dump\_accel

dump\_eos

err\_tol 1.e-3

energy\_smooth 0.2 0

set\_units stress kbar 1.e9

#-------regions----------

set\_region ball

material al2017

eos grun

end\_region

set\_region plate

material al6061

eos grun

thin

end\_region

#--------strength--------

strength\_model elas\_str\_hard

fracture

#---- model build follows----

begin\_region ball

part\_mult = 0.9

do\_sphere 0.4765

translate\_reg 0. 0.4765 0.

velocity\_reg 0. -6.62e5 0.

begin\_region plate

part\_mult = 0.1

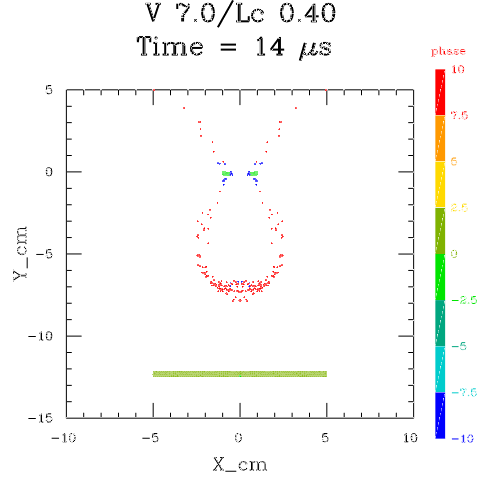
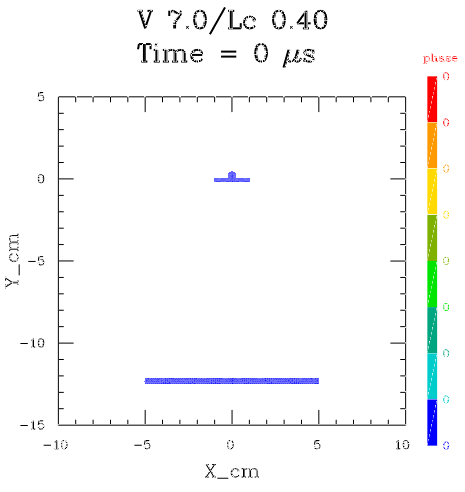
do\_block 2. 0.0465 2.

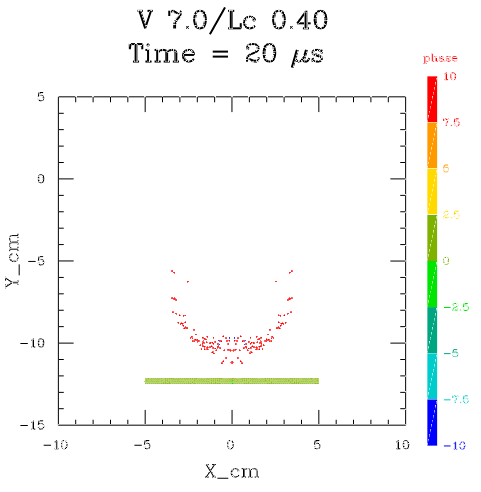
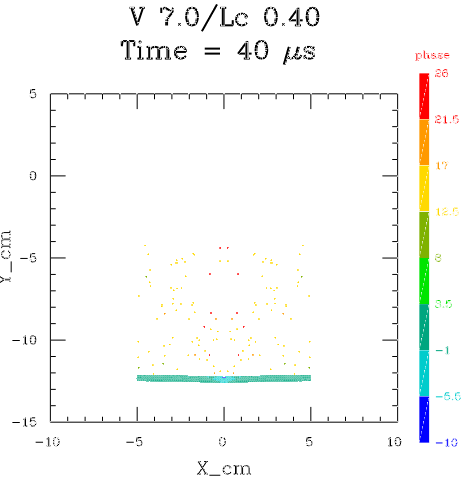
translate\_reg 0.0 -0.0232 0.

### Plate and Bulkhead Impact – 2D

The success or failure of the previous case depends on the behavior of the debris cloud at much later times, and how it interacts with a pressure bulkhead that is being protected. This impact is similar to the previous case, with a sphere diameter of 4 mm and a velocity of 7 km/s. All materials are aluminum, but different alloys, as used in the experiments. In this case, much less resolution is used in the impacting sphere, but the calculation is carried to a later time during which the debris cloud develops completely. The cloud then impacts on a second plate, representing the bulkhead. The experiment for this case showed some surface damage to the bulkhead, but no penetration.

This case is coded in the more modern setup style than the previous case. It includes parameter specification using local variables, parameter inclusion in the problem title strings, input and output unit conversion factors, plot dumps, and new material specifications. In addition, it includes several new techniques for increasing the accuracy of this type of simulation, including the “thin” flag for thin layers, outer absorb boundaries to limit the volume of the calculation and drop material that has been “left behind” from the first impact, and the “delay\_regions” command, that removes the bulkhead plate from the calculation (after the initial dumps) until material has moved into its vicinity. These options greatly reduce the run time without affecting the result. Use of local variable names, also greatly increase the readability of the code and make it very simple to change variables, dimensionality, run resolution or problem design.





File name is “double”. Run time is 10.4 s.

#====double plate test case====

# modified 6/24/2004 - rfs

#---set probelm params---------------------

Spd = 7.0

Balldiam = 0.40

#------------------------------------------

#---construct title strings---

decimals 1; $str = "V "; str\_add $str @Spd

problem\_title $str

decimals 2; $str = "Lc "; str\_add $str @Balldiam

run\_title $str

#----------options------

# debug\_part = 1

#-----------------------

#----conversion factors----

Cm = 1

In = 2.54

Ft = 12\*In

Psi = 6.895e4

Ksi = 1.e3\*Psi

Bar = 1.e6

Kbar = 1.e9

Lbf = 4.4482e5

Lbm = 453.59

Usec = 1.e-6

Msec = 1.e-3

Ftps = Ft

Kmps = 1.e5

Lbft3 = 1.6018e-2

Ftlb = 1.3558e7

Hz = 2\*pi

// output units

set\_units location cm Cm

set\_units velocity ft/s Ftps

set\_units time ms Msec

set\_units density lb/ft3 Lbft3

set\_units stress psi Psi

set\_units probe\_stress psi Psi

set\_units bdry\_force lb Lbf

set\_units energy ftlb Ftlb

#------Setup variables-------

Ballspd = Spd\*Kmps

Wallthk = 0.16

Wallxz = 2.0

Wall2thk = 0.32

if Spd>5 // narrower at lower speeds

Wall2xz = 10

else

Wall2xz = 5

end\_if

Standoff = 12.

#----2D settings---

dimension = 2

nparticles = 1500

space\_adjust = 1.1

cylindrical

max\_time = 40.e-6

restart\_dumps = 5

plot\_dumps = 20

hist\_dumps = 100

err\_tol = 0.01

energy\_smooth 0.2 0

#---------set delay here----------------

Delay = 1-(Spd-7)/15

delay\_regions 3 0.8\*Standoff/Ballspd

#---------------------------------------

# this turns on the vaporization temp

# correction for rho > rho0

# based on Al, Cu and Zn data

eos\_den\_dep

# this randomizes the particles slightly

pert\_size = 5

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Rball = Balldiam/2.

Ballvol = pi\*Rball^2

Wall1vol = Wallxz\*Wallthk

Wall2vol = Wall2xz\*Wall2thk

Totvol = Ballvol+Wall1vol+Wall2vol

#------materials------

# Al2024-T81

add\_material al2024

rho\_0 = 2.78

a\_mol = 27

a\_atm = 26.95

z\_atm = 13

ion\_en = 5.96

gamma\_G = 2.1

s\_shock = 1.55

gamma\_mol = 1.6667

cs\_0 = 5.38e5

cv\_0 = 0.904e7

cv\_liq = 1.e7

tmelt = 916

hmelt = 3.9e9

tvap = 2500 //??

hvap = 1.e11 //??

ey = 72.4e10

pr = 0.33

sy = 450.e7

st = 485.e7

em = 0.06 // reduced 3/13

br = 0.3

end\_material

# Al 2219-T87

add\_material al2219

rho\_0 = 2.84

a\_mol = 27

a\_atm = 26.95

z\_atm = 13

ion\_en = 5.96

gamma\_G = 2.1 //?

s\_shock = 1.55 //?

gamma\_mol = 1.6667

cs\_0 = 5.38e5

cv\_0 = 0.864e7

cv\_liq = 1.e7

tmelt = 916

hmelt = 3.9e9

tvap = 2500 //??

hvap = 1.e11 //??

thmcon = 1.20e7

ey = 72.e10

pr = 0.33

sy = 395.e7

st = 475.e7

em = 0.10

br = 0.3

end\_material

#--------strength--------

strength\_model high\_str\_rate

fracture

#------boundaries-----

set\_boundary top

direction y

location -4

side high

type absorb

time 10/Ballspd

end\_boundary

set\_boundary outer

direction x

location Wall2xz/2

side high

type absorb

end\_boundary

#-------regions----------

// ties to Palmieri et al. in HVIS2000

set\_region ball

material al2024

eos grun

end\_region

set\_region bumper

material al2024

eos grun

thin // 2D only

end\_region

set\_region backwall

material al2219

eos grun

thin // 2D only

end\_region

#---- model build follows----

get\_reg\_density ball Bden

get\_reg\_density bumper Bprden

get\_reg\_density backwall Bkwlden

begin\_region ball

part\_mult = 1.5\*Ballvol/Totvol

do\_sphere Rball

mass\_reg Bden\*Ballvol/2

translate\_reg 0. 1.1\*Rball 0.

velocity\_reg 0. -Ballspd 0.

begin\_region bumper

part\_mult = 1.5\*Wall1vol/Totvol

do\_block Wallxz Wallthk Wallxz

mass\_reg Bprden\*Wall1vol/2

translate\_reg 0. -Wallthk/2 0.

begin\_region backwall

part\_mult = Wall2vol/Totvol

do\_block Wall2xz Wall2thk Wall2xz

mass\_reg Bkwlden\*Wall2vol/2

translate\_reg 0. -2\*Wallthk-Standoff 0

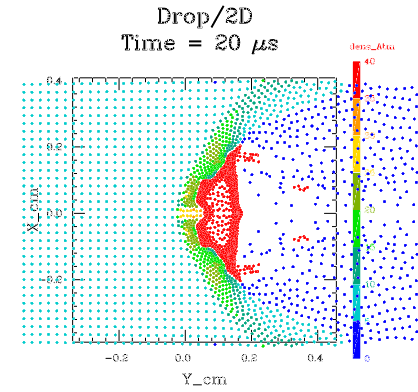
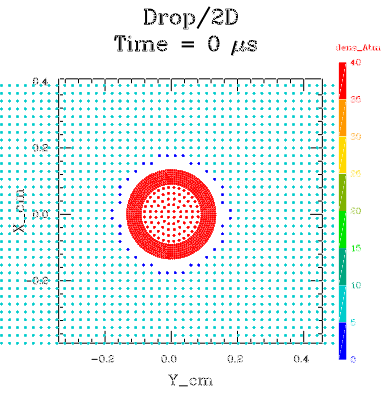
### Water Drop in Mach 5 Wind Tunnel – 2D

Development of a robust wind tunnel simulation has proven to be elusive for SPH codes. The natural idea of introducing new particles at the inflow plane has been found to produce unacceptable unevenness in the flow. SPHC has a wind tunnel option that has proven to be stable and accurate for most problems. This implementation uses reciprocating pistons at the entry and exit planes. A buffer of material is inserted into the flow at the entry for each cycle, which are computed automatically by the code during the run. The new material inserted each cycle merges with the flow from the previous cycle. This technique works perfectly with supersonic flow, and usually gives an adequate model even at subsonic speeds, at least for early flow times.

Another challenging problem for any code is the interaction of a gas flow field and water droplets. The flow will change the shape and movement of the drop, which, in turn, will affect the flow field. This validation test includes the use of an advanced Van der Waals equation of state for the drop, use of the factor “mu” (see User’s Guide) to model the surface tension of the drop, and multi-layer resolution in the drop to increase the resolution in the outer layer of the drop to resolve the ablation at the edge.

The first water drop validation case uses a Mach 5 flow speed in a 2D model with cylindrical geometry along the axis of the tunnel. This models a spherical drop in 2 dimennsions.

Beginning and ending plots, colored on density:



Run time is 19 sec. File name is “drop2D”, input file follows.

#====shock tube test case====

# 2D Water Drop in Mach 5 flow

# test case version

# soft reflect outer boundaries

# inflow and outflow boundaries

# rfs - 8/9/8

problem\_title "Drop"

run\_title "2D"

//timers

Atmden = 1.2928e-3

In = 2.54

Cm = 1

Cmps = 1

Ft = 12\*In

Ftps = Ft

Psi = 6.895e4

Gmpcc = 1

K = 1

set\_units density Atm Atmden

//set\_units velocity ft/s Ftps

set\_units velocity cm/s Cmps

set\_units location cm Cm

set\_units stress psi Psi

#---user variables--------

Drop\_rad = .135\*Cm

Vel = 5\*3.e4\*Cmps // Mach 5 for test case

Width = 0.8\*Cm // y direction (1.2)

Length = 1.0\*Cm // gas working region

// if the inflow cycling causes noise, increase the

// inflow length

Inflow\_length = Length/8 // inflow region

Shift = 0 // use this to move drop position

Density = 4\*0.001783\*Gmpcc // high den (post shock)

Temp = 357.\*K

quiet\_start // needed to get uniform initial densities

V1 = Length\*Width //\*Width

V2 = Inflow\_length\*Width //\*Width

Vdrop = pi\*sq(Drop\_rad) //5\*1.33\*pi\*cub(Drop\_rad)

Vtot = V1+2\*V2+Vdrop // inflow region should be counted twice

#-------------------------

dimension = 2

nparticles = 2000 // adjust as needed

# space\_adjust determines the max number of particles

# the block will expand in size due to the moving

# right boundary

# may need adjusting if long times are desired

# try values, and check the memory used

space\_adjust 1.5

max\_time = 20.e-6

restart\_dumps = 2

hist\_dumps = 20

plot\_dumps = 10

plot\_temp

plot\_vx

plot\_vy

plot\_press

plot\_weber

plot\_db

plot\_probes

pert\_size = 0.0

h\_inp = 1.

h\_vary = true

cylindrical

energy\_smooth

pmin = -5e6 // standard

#--------probes-----------

set\_probe fixed 0.2 0 0 2 // test in gas region

#---------boundaries-------

#---inflow boundary---

set\_boundary inflow

direction y

location = -Length/2-Shift-Inflow\_length

side low

velocity = Vel

buffer\_width = Inflow\_length

end\_boundary

#---outflow at right---

set\_boundary outflow

direction y

location = Length/2-Shift

side high

type = reflect2

buffer\_width = Inflow\_length

velocity = Vel

drift\_velocity = Vel/10 // allow drift

end\_boundary

#---tube walls---

set\_boundary top

direction x

location = Width/2

side high

type soft\_reflect

end\_boundary

#-------regions----------

set\_region drop

material h2o

eos moylan

mu = 1.e-3 // gives best surface tension effect

pressure = 4\*26.6\*Psi // equal to the gas pressure

end\_region

set\_region reg0 // this is the gas region with the drop

material pg

eos pg

gamma = 1.4

mu = 29.

density = Density

temp = Temp

end\_region

#---normally the inflow region matches reg0

# but, could be different to represent a shock, etc.

# the inflow region will regenerate as needed

set\_region inflow // inflow region

material pg

eos pg

gamma = 1.4

mu = 29.

density = Density

temp = Temp

end\_region

#---- model build follows----

begin\_region drop

part\_mult = 4\*2\*Vdrop/Vtot

do\_sphere Drop\_rad Drop\_rad/3

part\_mult 4\*0.2\*Vdrop/Vtot

do\_sphere 2\*Drop\_rad/3

begin\_region reg0

part\_mult = V1/Vtot

do\_block Width Length Width

velocity\_reg 0 Vel 0

translate\_reg 0 -Shift 0

trim\_reg 1.75

#---build inflow block---must be last one---

begin\_region inflow

part\_mult = V2/Vtot

do\_block Width Inflow\_length Width

velocity\_reg 0 Vel 0

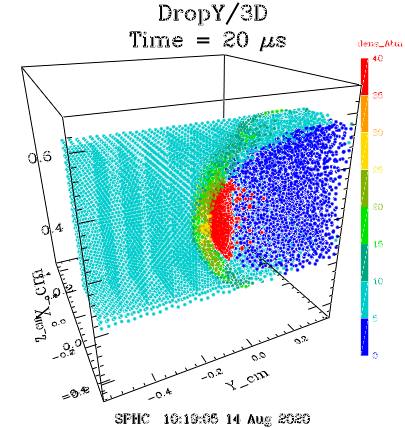
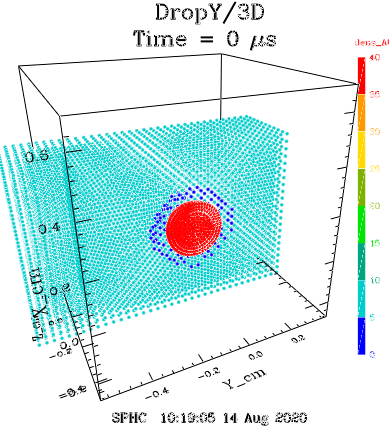
translate\_reg 0 -Length/2-Shift-Inflow\_length/2 0

### Water Drop / Rectangular Mach 5 Wind Tunnel– 3D

This is a 3D version of the previous test case. In this case the wind tunnel is modeled as rectangular in cross section, with double reflect symmetry conditions imposed to decrease the run time. For final results models, these conditions would normally be withdrawn and a longer. But more accurate, run would be the result.

Note that at time 0 the inflow buffer is shown beyond the inflow boundary (at the left) for debugging the setup. At later times this buffer is not included in the plot or restart dumps.

Start and end of the simulation is shown, colored on density. Rear half of the simulation shown. Compare with previous case.



During the run, recycle information will appear in the screen list as follows:

==>Recycle inflow boundary, time=1.91698e-05, Npart: 5611->6091

new buffer=6092-6571

==>Recycle outflow boundary, vel=150000\_cm/s, drift=15000\_cm/s, cycle=9.25926e-07

Run time for this case was 1.80 min. File name is “drop3D.inp”, listing follows.

#====shock tube test case====

# 3D Water Drop in Mach 5 flow

# test case version

# soft reflect outer boundaries

# inflow and outflow boundariesd

# rfs - 8/9/8

problem\_title "DropY"

run\_title "3D"

//timers

Atmden = 1.2928e-3

In = 2.54

Cm = 1

Cmps = 1

Ft = 12\*In

Ftps = Ft

Psi = 6.895e4

Gmpcc = 1

K = 1

set\_units density Atm Atmden

//set\_units velocity ft/s Ftps

set\_units velocity cm/s Cmps

set\_units location cm Cm

set\_units stress psi Psi

#---user variables--------

Drop\_rad = .135\*Cm

Vel = 5\*3.e4\*Cmps // Mach 5 for test case

Width = 0.8\*Cm // y direction (1.2)

Length = 1.0\*Cm // gas working region

// if the inflow cycling causes noise, increase the

// inflow length

Inflow\_length = Length/8 // inflow region

Shift = 0 // use this to move drop position

Density = 4\*0.001783\*Gmpcc // high den (post shock)

Temp = 357.\*K

quiet\_start // needed to get uniform initial densities

V1 = Length\*Width\*Width

V2 = Inflow\_length\*Width\*Width

Vdrop = 5\*1.33\*pi\*cub(Drop\_rad)

Vtot = V1+2\*V2+Vdrop // inflow region should be counted twice

#-------------------------

dimension = 3

nparticles = 10000 // adjust as needed

# space\_adjust determines the max number of particles

# the block will expand in size due to the moving

# right boundary

# may need adjusting if long times are desired

# try values, and check the memory used

space\_adjust 1.3

max\_time = 20.e-6

restart\_dumps = 2

hist\_dumps = 20

plot\_dumps = 10

plot\_temp

plot\_vx

plot\_vy

plot\_pressz

plot\_weber

pert\_size = 0.0

h\_inp = 1.

h\_vary = true

symmetry x

symmetry z

energy\_smooth

pmin = -5e6 // standard

#---------boundaries-------

#---inflow boundary---

set\_boundary inflow

direction y

location = -Length/2-Shift-Inflow\_length

side low

velocity = Vel

buffer\_width = Inflow\_length

end\_boundary

#---outflow at right---

set\_boundary outflow

direction y

location = Length/2-Shift

side high

type = reflect2

buffer\_width = Inflow\_length

velocity = Vel

drift\_velocity = Vel/10 // allow drift

end\_boundary

#---tube walls---

set\_boundary top

direction x

location = Width/2

side high

type soft\_reflect

end\_boundary

/\*

set\_boundary bottom

direction x

location = -Width/2

side low

type soft\_reflect

end\_boundary

\*/

set\_boundary front

direction z

location = Width/2

side high

type soft\_reflect

end\_boundary

/\*

set\_boundary back

direction z

location = -Width/2

side low

type soft\_reflect

end\_boundary

\*/

#-------regions----------

set\_region drop

material h2o

eos moylan

mu = 1.e-3 // best surface temsion

pressure = 4\*26.6\*Psi // equal to the gas pressure

end\_region

set\_region reg0 // this is the gas region with the drop

material pg

eos pg

gamma = 1.4

mu = 29.

density = Density

temp = Temp

end\_region

#---normally the inflow region matches reg0

# but, could be different to represent a shock, etc.

# the inflow region will regenerate as needed

set\_region inflow // inflow region

material pg

eos pg

gamma = 1.4

mu = 29.

density = Density

temp = Temp

end\_region

#---- model build follows----

// get exact sphere setup for testing, revert when done

begin\_region drop

part\_mult = 2\*Vdrop/Vtot

do\_sphere Drop\_rad Drop\_rad/3

part\_mult 0.2\*Vdrop/Vtot

do\_sphere 2\*Drop\_rad/3

begin\_region reg0

part\_mult = V1/Vtot

do\_block Width Length Width

velocity\_reg 0 Vel 0

translate\_reg 0 -Shift 0

trim\_reg 1.75

#---build inflow block---must be last one---

begin\_region inflow

part\_mult = V2/Vtot

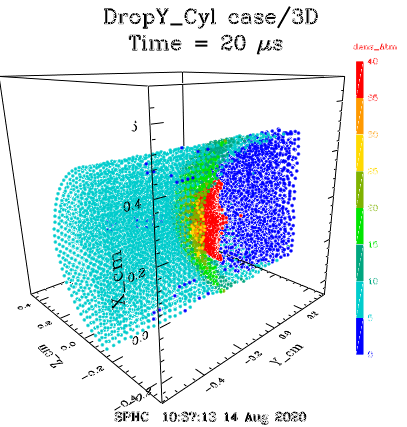
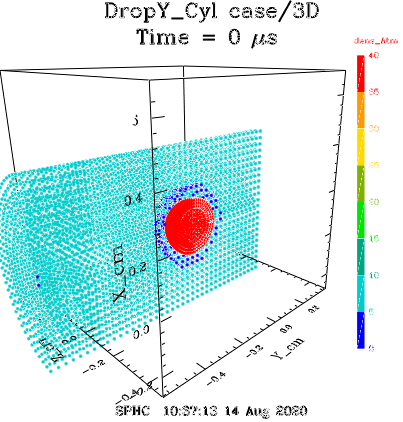
do\_block Width Inflow\_length Width

velocity\_reg 0 Vel 0

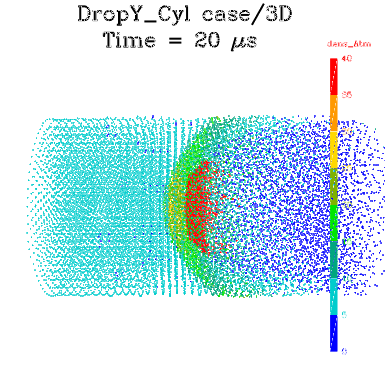
translate\_reg 0 -Length/2-Shift-Inflow\_length/2 0

### Water Drop / Cylindrical Mach 5 Wind Tunnel – 3D

This run duplicates the previous test case, but in a cylindrical wind tunnel to test the variation in the geometry and boundary conditions and their effect on the result. This is the same geometry as the 2D test case. Beginning and end of the simulation is shown below, colored on density.



Another view, compare to the previous runs.



This run takes 2.52 min. Run file is “drop3d\_cyl.inp”. Listing follows.

#====shock tube test case====

# 3D Water Drop in Mach 5 flow

# test case version

# soft reflect outer boundaries

# inflow and outflow boundaries

#cylindrical version of inflow

# rfs - 8/9/8 jr 4 23 18

problem\_title "DropY\_Cyl case"

run\_title "3D"

//timers

Atmden = 1.2928e-3

In = 2.54

Cm = 1

Cmps = 1

Ft = 12\*In

Ftps = Ft

Psi = 6.895e4

Gmpcc = 1

K = 1

set\_units density Atm Atmden

//set\_units velocity ft/s Ftps

set\_units velocity cm/s Cmps

set\_units location cm Cm

set\_units stress psi Psi

#---user variables--------

Drop\_rad = .135\*Cm

Vel = 5\*3.e4\*Cmps // Mach 5 for test case

Width = 0.8\*Cm // y direction (1.2)

Length = 1.0\*Cm // gas working region

// if the inflow cycling causes noise, increase the

// inflow length

Inflow\_length = Length/8 // inflow region

Shift = 0 // use this to move drop position

Density = 4\*0.001783\*Gmpcc // high den (post shock)

Temp = 357.\*K

quiet\_start // needed to get uniform initial densities

V1 = pi\*Length\*Width\*Width/4

V2 = pi\*Inflow\_length\*Width\*Width/4

Vdrop = 5\*1.33\*pi\*cub(Drop\_rad)

Vtot = V1+2\*V2+Vdrop // inflow region should be counted twice

#-------------------------

dimension = 3

nparticles = 10000 // adjust as needed

# space\_adjust determines the max number of particles

# the block will expand in size due to the moving

# right boundary

# may need adjusting if long times are desired

# try values, and check the memory used

space\_adjust 1.3

max\_time = 20.e-6

restart\_dumps = 2

hist\_dumps = 20

plot\_dumps = 10

plot\_temp

plot\_vx

plot\_vy

plot\_pressz

plot\_weber

pert\_size = 0.0

h\_inp = 1.

h\_vary = true

symmetry x

symmetry z

energy\_smooth

pmin = -5e6 // standard

#---------boundaries-------

#---inflow boundary---

set\_boundary inflow

direction y

location = -Length/2-Shift-Inflow\_length

side low

velocity = Vel

buffer\_width = Inflow\_length

end\_boundary

#---outflow at right---

set\_boundary outflow

direction y

location = Length/2-Shift

side high

type = reflect2

buffer\_width = Inflow\_length

velocity = Vel

drift\_velocity = Vel/10 // allow drift

end\_boundary

#---tube walls---

set\_boundary outer

direction ry

location = Width/2

side high

type soft\_reflect

end\_boundary

#-------regions----------

set\_region drop

material h2o

eos moylan

mu = 1.e-3 // best surface tension

pressure = 4\*26.6\*Psi // equal to the gas pressure

end\_region

set\_region reg0 // this is the gas region with the drop

material pg

eos pg

gamma = 1.4

mu = 29.

density = Density

temp = Temp

end\_region

#---normally the inflow region matches reg0

# but, could be different to represent a shock, etc.

# the inflow region will regenerate as needed

set\_region inflow // inflow region

material pg

eos pg

gamma = 1.4

mu = 29.

density = Density

temp = Temp

end\_region

#---- model build follows----

// get exact sphere setup for testing, revert when done

begin\_region drop

part\_mult = 2\*Vdrop/Vtot

do\_sphere Drop\_rad Drop\_rad/3

part\_mult 0.2\*Vdrop/Vtot

do\_sphere 2\*Drop\_rad/3

begin\_region reg0

part\_mult = V1/Vtot

do\_cylinder Width/2 Length

velocity\_reg 0 Vel 0

translate\_reg 0 -Shift 0

trim\_reg 1.75

#---build inflow block---must be last one---

begin\_region inflow

part\_mult = V2/Vtot

do\_cylinder Width/2 Inflow\_length

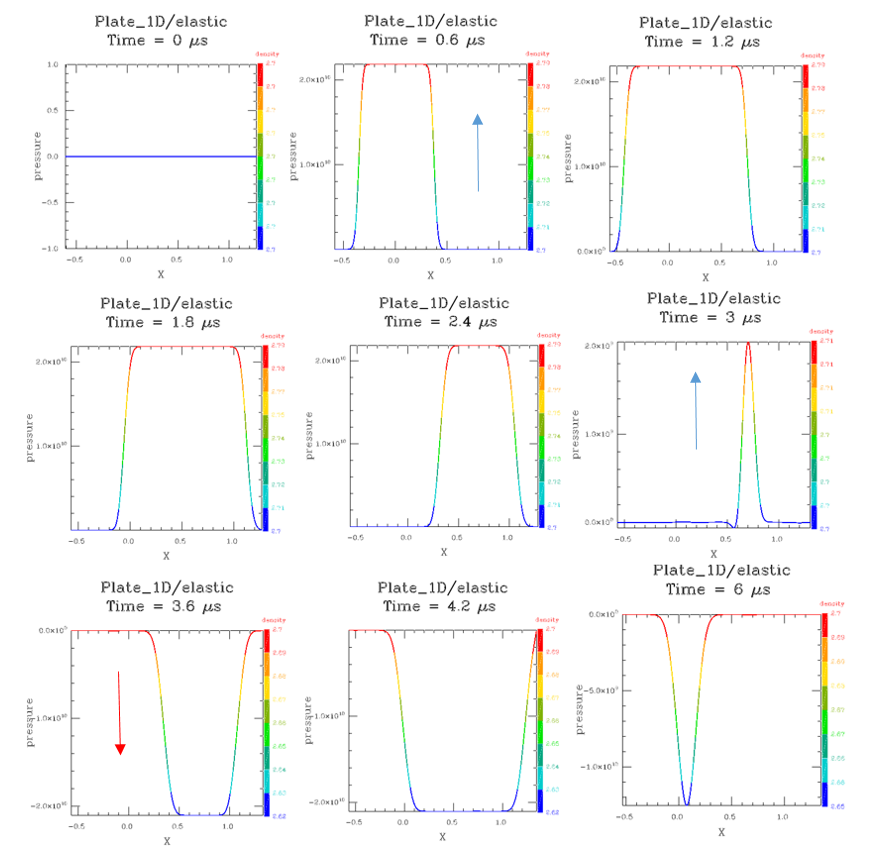
velocity\_reg 0 Vel 0

translate\_reg 0 -Length/2-Shift-Inflow\_length/2 0

### Flyer Plate Impact – 1D

This is a standard aluminum “Plate-on-plate” impact experiment. A moving plate (velocity 0.39 km/s) impacts a stationary plate. The moving plate is half the thickness of the target plate. This test uses the simplest setup - a 1 dimensional run with infinite plate widths, so, no edge effects. The purpose of the test is to model the shock wave caused by the impact, and its subsequent reflections and merges over time. In particular, at 3 s after the two shocks have reflected from the ends of the plate, they meet at the center of the thicker plate and become a strong rarefaction wave. This results in the SPH “tensile instability” where the pressure becomes violently unstable. This is a feature of all cell-centered codes. The new VSP (Virtual Stress Point) used in SPHC eliminates this instability, and this test verifies this.

The plots below show the run of pressure at various times, note the behavior after 3 s.



File name for this test is “plate1D.inp”, run time is 2.11 s.

**#====flyer plate test case====**

**problem\_title "Plate\_1D"**

**run\_title "elastic"**

**dimension = 1**

**nparticles = 500**

**//space\_adjust = 1.02**

**max\_time = 6.e-6**

**restart\_dumps = 30**

**hist\_dumps = 60**

**pert\_size = 0. // no rand**

**pmin = -1.e11 // no pmin**

**dump\_accel**

**dump\_eos**

**#-----user variables-----**

**Veloc = 0.39e5**

**Thick1 = 0.608 // left slab**

**Thick2 = 1.27 // right slab**

**#-------strength-------**

**strength\_model elastic**

**#-------regions----------**

**set\_region "moving"**

**material al**

**//eos usup**

**eos linear**

**end\_region**

**set\_region "fixed"**

**material al**

**//eos usup**

**eos linear**

**end\_region**

**#-------data probes--------**

**set\_probe moving -Thick1/2**

**set\_probe moving 0.**

**set\_probe moving Thick2/2**

**#---- model build follows----**

**begin\_region "moving"**

**part\_mult = Thick1/(Thick1+Thick2)**

**do\_block Thick1 1.0 1.0**

**translate\_reg -Thick1/2 0. 0.**

**velocity\_reg Veloc 0 0**

**begin\_region "fixed"**

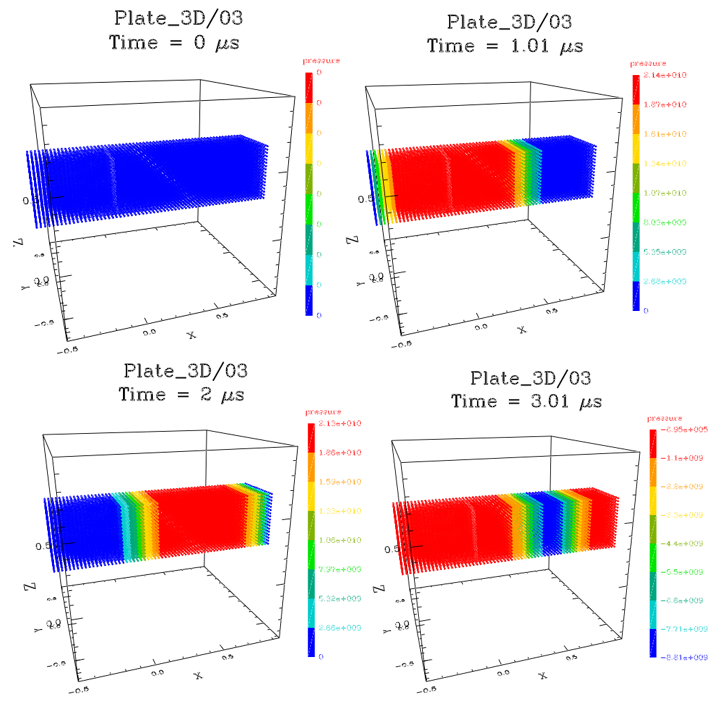
**part\_mult = Thick2/(Thick1+Thick2)**

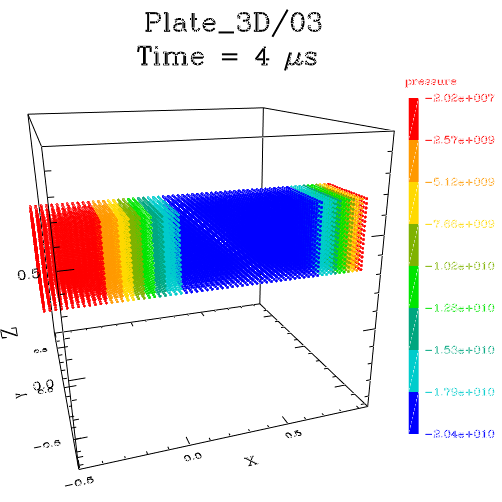
**do\_block Thick2 1.0 1.0**

**translate\_reg Thick2/2 0. 0.**

### Flyer Plate Impact – 3D

This is the 3D version of the previous test case. Stability in the rarefaction is preserved, but the shocks are smoothed out a bit relative to the 1D run. Only 20,000 particles are used for this run, which is a very low number for a 3D run. This is to reduce run time for the test. For an application, at least 200,000 particles would be used, and the shocks would be much better defined.





File name for this test is “plate3D.inp”, run time is 2.9 min.

#====3D flyer plate test case====

problem\_title "Plate\_3D"

run\_title "03"

dimension = 3

nparticles = 20000

space\_adjust = 1.05

max\_time = 1.e-6

restart\_dumps = 10

hist\_dumps = 50

pert\_size = 0. // no rand

h\_inp = 1.

pmin = -1.e11 // no pmin

dump\_accel

dump\_eos

#-----user variables-----

Veloc = 0.39e5

Thick1 = 0.608 // left slab

Thick2 = 1.27 // right slab

#-------strength-------

strength\_model elastic //off for inst

#-------regions----------

set\_region "moving"

material al

//eos usup

eos linear

end\_region

set\_region "fixed"

material al

//eos usup

eos linear

end\_region

#-------data probes--------

set\_probe moving -.3

set\_probe moving 0.

set\_probe moving 0.6

#-------boundaries--------

symmetry y

symmetry z

set\_boundary top

location 0.5

direction y

side high

type reflect

end\_boundary

set\_boundary front

location 0.5

direction z

side high

type reflect

end\_boundary

#---- model build follows----

begin\_region "moving"

part\_mult = Thick1/(Thick1+Thick2)

do\_block Thick1 1.0 1.0

translate\_reg -Thick1/2 0. 0.

velocity\_reg Veloc 0 0

begin\_region "fixed"

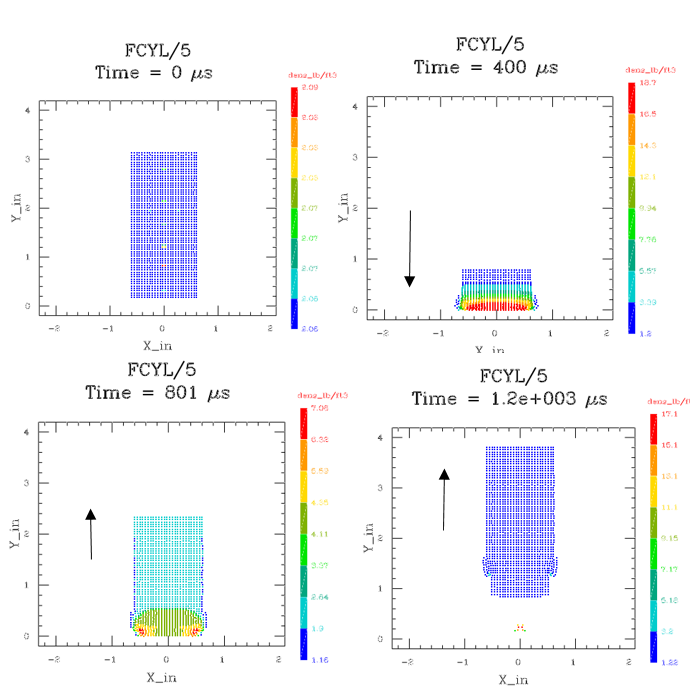
part\_mult = Thick2/(Thick1+Thick2)

do\_block Thick2 1.0 1.0

translate\_reg Thick2/2 0. 0.

### Stiff Foam Cylinder Impact – 2D

This is a test of a BX250 foam cylinder impacting a fixed boundary at 700 ft/s. This case was run at NASA Glenn laboratory and showed in most cases extreme compression of the foam, followed be a near-elastic rebound at about half the initial impact velocity, about the initial length of the foam cylinder, and with only minor damage at the end of the cylinder. This test is a simplified version of this case. The main modeling challenge is to see if the spherical SPH particles could successfully model the compressive phase, which would normally be handled in other codes by a distortion of a grid, and then return the shape to its original form. The result is shown below, colored on density. The extreme compression length and final velocity of the foam are close the the experimental results.



Input file is named “Foam\_Cyl.inp”. Run time is 19.38 s. Listing below.

#====foam cylinder impact====

#---uses 3D foam model---

problem\_title "FCYL"

run\_title "5"

debug\_part = 20

#----conversion factors----

In = 2.54

Ft = 12\*In

Psi = 6.895e4

Ksi = 1.e3\*Psi

Bar = 1.e6

Kbar = 1.e9

Lbf = 4.4482e5

Lbm = 453.59

Usec = 1.e-6

Msec = 1.e-3

Ftps = Ft

Kmps = 1.e5

Lbft3 = 1.6018e-2

Ftlb = 1.3558e7

Hz = 2\*pi

// output units

set\_units location in In

set\_units velocity ft/s Ftps

set\_units time ms Msec

set\_units density lb/ft3 Lbft3

set\_units stress psi Psi

set\_units probe\_stress psi Psi

set\_units bdry\_force lb Lbf

set\_units energy ftlb Ftlb

#----basic 2D settings---

dimension = 2

nparticles = 1000

space\_adjust = 1.05

cylindrical

max\_time = 1.2\*Msec

restart\_dumps = 12

hist\_dumps = 60

err\_tol 1.e-2

energy\_smooth .1

#----setup----

FoamDiam = 1.25\*In

FoamLen = 3\*In

FoamDensity = 2.06\*Lbft3 // GRC test value

FoamVel = 700\*Ftps

#----strength model----

strength\_model elas\_str\_hard

#---wall boundary---

set\_boundary wall

direction y

slip

track\_force

end\_boundary

#---materials---

add\_material bx250

//rho\_0 = 1.15 // use .038

rho\_0 = 0.35

a\_mol = 100 //??

a\_atm = 20 //??

z\_atm = 10 //??

ion\_en = 10 //??

gamma\_G = 0.5 // lanl value

s\_shock = 0.75

gamma\_mol = 1.333

//cs\_0 = 1.e5 //??

cs\_0 = 5.e4 // GRC bounce

cv\_0 = 1.5e6

cv\_liq = 1.e6

tmelt = 600 //??

hmelt = 1.e8 //??

tvap = 1000 //??

hvap = 1.e9 //??

//ey = 5.0e8

//ey = 7.322e7 // from sr 4/4

sy = 5.0e6 //??

pr = 0.07 // new result

//st = 5.2e6

em = 0.145 // new

br = 0.5

ey = 2.6\*Ksi // ED30 numbers

sy = 60.\*Psi

st = 80.\*Psi

pr = 0.07 // BX250

em = 0.12 // ED30

end\_material

#-------regions----------

set\_region foam

material bx250

density FoamDensity

pressure = 1\*Bar //pore pressure

eos crush

elastic\_crush

p\_elastic = 13\*Psi // from fit

p\_crush = 95\*Psi

av\_alpha = 3.5 // dissipation

av\_beta = 3.5

strength\_model elastic // bx250

//fracture // not bad either way

strength\_mode xyz 1 3 1 // aniso here

end\_region

#---- model build follows----

begin\_region foam

part\_mult = 0.9

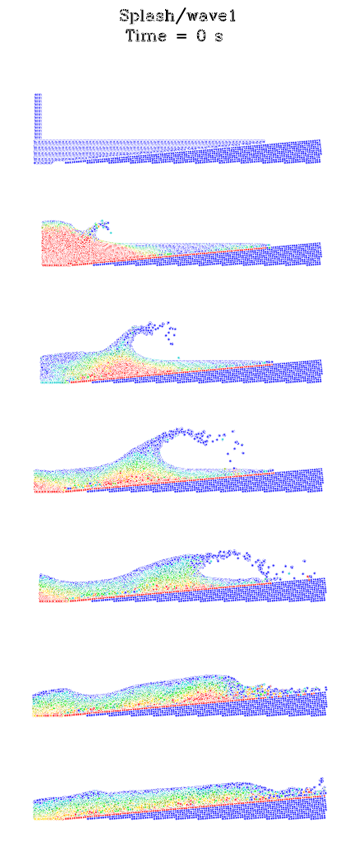
do\_cylinder FoamDiam/2 FoamLen

translate\_reg 0. 1.1\*FoamLen/2 0.

velocity\_reg 0. -FoamVel 0.

### Breaking Water Wave – 2D

This is a simple model of a wave breking on a beach. A layer of water is created over an inclined “frozen” block “beach”. A large block of water is then added on top of the left side of the initial layer and allowed to flow. The result is a breaking wave, shown here colored on pressure. Plot interval is ½ second, and the peak pressure is 5 psi.



Input file name is “wave1.inp”, run time is 2.02 min, listing follows.

#====Sphere Water Impact Test==========================

/\* breaking wave \*/

problem\_title "Splash"

run\_title "wave1"

#----conversion factors----

read\_file units.inp

// output units

set\_units location m Mtr

set\_units velocity ft/s Ftps

set\_units time s 1

set\_units stress psi Psi

#----basic settings--------------------------------------

Velocity = 10\*Mps

dimension 2

npart = 10000

space\_adjust = 1.0

#---set dumps here---

max\_time = 3

restart\_dumps = 3

plot\_dumps = 30

plot\_press

plot\_flow

hist\_dumps = 80

quiet\_start

pmin = -5.e6 // surface tension

err\_tol .01

energy\_smooth .1

// viscosity params

balsara // reduce shear

gravity 0 -Grav 0

slip\_regions 0.1 // friction

#----user params----------------------------------------

// sphere dimensions

Lslug = 4.5\*Mtr

Hslug = 6\*Mtr

// Wave dimensions

Height = 3\*Mtr // complete box height

Length = 30\*Mtr // wave box length

// Volumes

Vslug = sq(Lslug)

Vwave = Height\*Length

Vt = Vslug+1.5\*Vwave

#------boundaries-----------------------------

set\_boundary left

location -Length/2

direction x

side low

type reflect

end\_boundary

set\_boundary bottom

location -Height

direction y

side low

type reflect

end\_boundary

#-------regions-------------------------------------------

set\_region ocean

material h2o

eos water

mu = 1.e-3 // <--- for time step adjustment...

pmin = -5.e6 // surface tension

end\_region

set\_region slug

material h2o

eos water

mu = 1.e-3 // <--- for time step adjustment...

pmin = -5.e6 // surface tension

end\_region

set\_region beach

material nylon

eos grun

frozen

end\_region

#---- model build follows---------------------------------

begin\_region ocean

part\_mult = Vwave/Vt

do\_block Length Height 0

translate\_reg 0 -Height/2 0.

begin\_region slug

part\_mult = Vslug/Vt

do\_block Lslug Hslug 0

translate\_reg -Length/2+Lslug/2 Hslug/2 0

velocity\_reg 5\*Mps -5\*Mps 0

Angle = (atan(Length/Height)/Deg-90)

begin\_region beach

part\_mult = 0.3\*Vwave/Vt

do\_block 1.2\*Length Height 0

translate\_reg .125\*Length 0 0

rotate\_reg 0 0 -Angle

translate\_reg -Height\*sin(Angle\*Deg)+5\*Mtr -Height 0.

make\_room\_reg 2

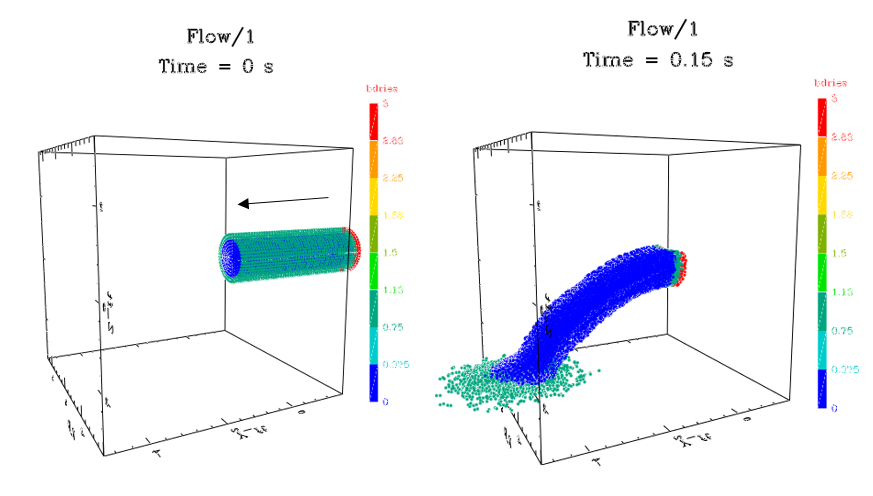
delete\_box -20\*Mtr 20\*Mtr -10\*Mtr -Height 0 0

### Flow from a Pipe - 3D

This case models water flowing in a pipe at 10 m/s. The pipe is modeled as a boundary condition that ends at the origin. The water is pushed through the pipe by a moving piston boundary at the rear end. After leaving the pipe, the water flows freely under a downward gravity force (enhanced here by x10). After falling 3 ft, the waterfall encounters a floor boundary and splashes. The entire run uses only a single region plus three specialized boundary conditions.

This case illustrates a short-cut for handling all possible sets of units – the file units.inp (listed in [Section 3.w](#_Units_Conversions_File)) is input to the run using a “read\_file” command at the top of the setup file.

Starting and ending snapshots, colored on number of adjacent boundaries.



Run time for this case is 3.01 min, file name is “flow.inp”.

#====Water flow/impact=======================

problem\_title "Flow"

run\_title "1"

#---Flow of water from a pipe---

read\_file Units.inp

// output units

set\_units location ft Ft

set\_units velocity ft/s Ftps

set\_units time ms Msec

//set\_units density lb/ft3 Lbmft3

set\_units stress psi Psi

set\_units probe\_stress psi Psi

set\_units bdry\_force lb Lbf

set\_units energy ftlb Ftlb

set\_units rotation Hz Hz

set\_units accel g Gearth

#----basic settings-----------------------------------

dimension = 3

nparticles = 12000

space\_adjust = 1.03

max\_time = 0.15

restart\_dumps = 5

plot\_dumps = 15

hist\_dumps = 30

plot\_press

plot\_flow

plot\_accel

plot\_v

plot\_vx

plot\_vy

plot\_vz

plot\_xyz0

plot\_bdries

quiet\_start

p\_offset

pmin = -5.e6 //surface tension

pert\_size = 0.1 //should drop!!!

energy\_smooth //was .1, left blank as in bobs fine file

err\_tol 0.01

no\_boundary\_warnings // for front nozzle

// viscosity params

balsara // reduce shear

gravity 0. 0. -10\*Gearth //add acceleration due to gravity

#----user params-------------------------------------

Pipe\_len = 5\*Ft

Runout = 4\*In // front pipe extension

#------boundaries--------------------------------------

set\_boundary pipe //to contain the water

location 10\*In

direction ry

side high

limit y 0 Pipe\_len+Runout

end\_boundary

set\_boundary piston // rear end of pipe

direction y

side high

location Pipe\_len+Runout

velocity -10\*Mtrps

limit ry 0 10\*In

end\_boundary

set\_boundary floor

direction z

side low

location -3\*Ft

end\_boundary

#-------regions----------------------------------------

set\_region water

material h2o

eos water

mu = 1.e-3 // surface tension

pmin = -5.e6 // surface tension

end\_region

#---- model build follows----------------------------

begin\_region water

set\_parts 10000

do\_cylinder 10\*In Pipe\_len

translate\_reg 0. Pipe\_len/2+Runout 0.

velocity\_reg 0 -10\*Mtrps 0.

### Units Conversions File

This file lists all of the currently used unit definitions. The numerical values are the factors needed to convert to native SPHC cgs units. All variables are normal user variables in SPHC, available to all codes, and are declared “constant”, to prevent redefinition by mistake in local code. In addition to the unit conversion factors, a list of useful mathematical and physical constants are provided.

File name is “units.inp”.

#----SPHC conversion factors----2/16 version---

# SPHC is cgs

# SI = SI units

# Br = British engineering units

constant // fix all values

#---constants---

Rgas = 8.317e7 // gas constant

Clight = 2.9979e10

Atmden = 1.2928e-3 // STP density

AU = 1.49597871e13 // astronomical unit

Msol = 1.9891e33 // solar values

Tesol = 5778

Lsol = 3.846e33

Rsol = 6.955e10

Gsol = 27444.1

Gconst = 6.674e-8 // grav constant

SIG = 5.6690e-05 // erg/cm2/deg^4/sec

N0 = 6.0232e23 // particles/mole

Gearth = 980.665 // earth grav

#--length = l

Cm = 1

Mm = 0.1

Um = 1.e-4

In = 2.54

Mil = .001\*In

Ft = 30.48 // Br

Mtr = 100 // SI

Km = 1.e5

Mi = 1.609344e5

Au = 1.4960e13

Ly = 9.4605e17

Pc = 3.0857e18

#--area = l^2

Cm2 = 1

Mm2 = 0.01

Mtr2 = 1.e4 // SI

In2 = 6.4516

Ft2 = 929.03 // Br

Km2 = 1.e10

Ba = 1.e-24

#--volume = l^3

Cm3 = 1

Mm3 = 0.001

Ltr = 1.e3

Qt = 946.35

Mtr3 = 1.e6 // SI

In3 = 16.387

Ft3 = 2.8317e4 // Br

Ozfl = 29.574

Gal = 3785.412

Bbl = 42\*Gal // oil barrel

#--time = t

Sec = 1 // SI, Br

Msec = 1.e-3

Usec = 1.e-6

Nsec = 1.e-9

Min = 60

Hr = 3600

Day = 86400

Yr = 31557600

Shake = 1.e-8

#--speed = l/t

Cmps = 1

Mtrps = 100 // SI

Kmps = 1.e5

Ftps = 30.48 // Br

Inps = 2.54

Mph = 44.70

Mps = Mph\*60\*60

#--accel = l/t^2

Cmps2 = 1

Mtrps2 = 100 // SI

Ftps2 = 30.48 // Br

Inps2 = 2.54

Grav = Gearth // old version

#--mass = m

Gm = 1

Kg = 1.e3 // SI

Mg = 1.e-3

Ug = 1.e-6

Lbm = 453.59237

Slug = Lbm\*Gearth/Ft // Br

BTon = 9.0718e5 // 2000 lb

Ton = 1.e6 // metric ton

Oz = 28.349523125

Grain = Lbm/7000

#--force = ml/t^2

Dyne = 1

Lbf = 4.4482e5 // Br

Kip = 1.e3\*Lbf

Gf = 980.67

Ntn = 1.e5 // SI

#--pressure = m/lt^2

Dynecm2 = 1

Bar = 1.e6

Kbar = 1.e9

Mbar = 1.e12

Psi = 6.895e4

Ksi = 6.895e7

Msi = 6.895e10

Pa = 10 // Nt/m2 SI

KPa = 1.e4

MPa = 1.e7

Mpa = MPa

GPa = 1.e10

Lbfft2 = 478.8 // Br

Lbfin2 = 68947.2

#--density = m/l^3

Gmpcc = 1

Lbmft3 = 1.6018e-2

Slgft3 = 0.515379 // Br

Lbmin3 = 27.68

Slgin3 = Lbmin3\*Gearth/Ft

Kgm3 = 1.e-3 // SI

#--areal density = m/l^2

Kgm2 = 0.10 // SI

Lbmft2 = 0.48824

Lbmin2 = 70.30696

Slgft2 = 15.7089 // Br

Slgin2 = 2262.08

#--energy = ml^2/t^2

Erg = 1

Joule = 1.e7 // SI

KJ = 1.e3\*Joule

MJ = 1.e3\*KJ

Cal = 4.1868e7

KCal = 1.e3\*Cal

Ftlb = 1.3558e7 // Br

Btu = 1.0551e10

Ev = 1.6022e-12

Kev = 1.e3\*Ev

Mev = 1.e6\*Ev

Kton = 4.2e19

Jerk = 1.e16

#--temperature--

DegK = 1 // SI

DegR = 5/9

DegF = 5/9 // ADD 255.372 Br

DegF\_shift = 225.372

DegC = 1 // ADD 273.15

DegC\_shift = 273.15

T\_Ev = 11604.505

#--power = ml^2/t^3

Ergps = 1

Watt = 1.e7 // SI

Hp = 7.457e9 // 550 Ftlbps

Ftlbps = 1.3558e7 // Br

#--frequency = t^-1

Radps = 1

Hz = 2\*pi // SI

#---angles

Rad = 1 // SI

Deg = pi/180

#---Miscl

Mpg = 42.515

constant false

## Applications

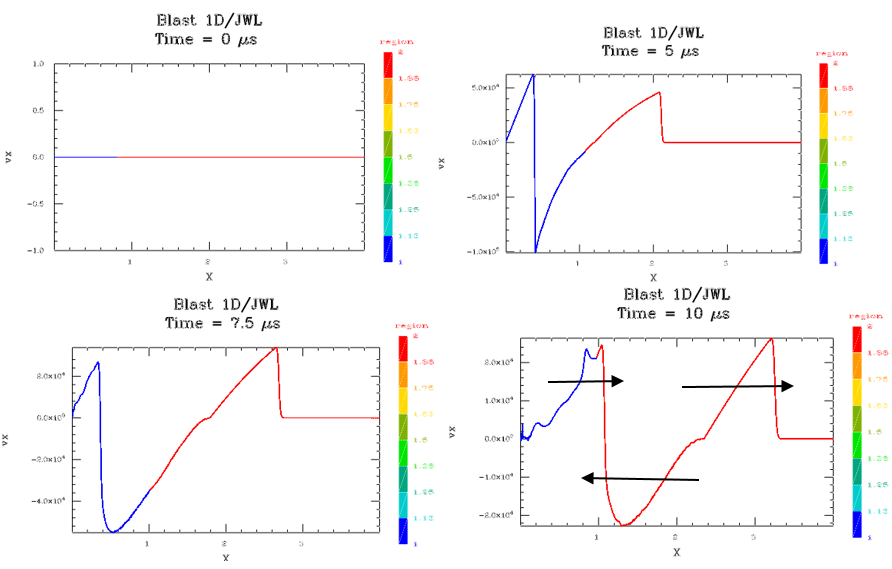
This section shows some examples of how the simple test cases can be modified to use on a more realistic scenario, and how the results are affected.

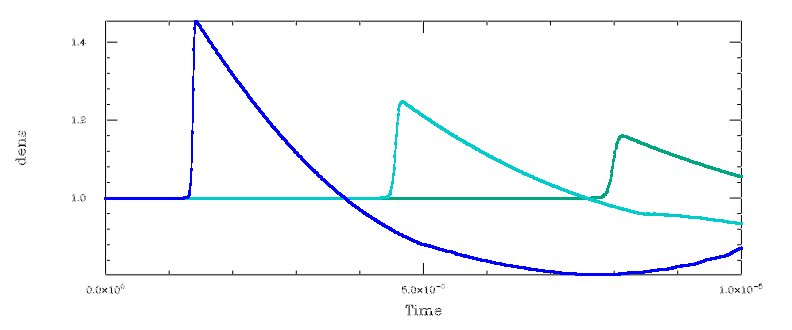
### Explosion

Any explosive scenario will generate a blast wave and will tend to evolve to something resembling the blast wave cases discussed above in its outer regions. Realistic cases will differ because of the lower (i.e. finite) energy of the blast, and the more extended size of the initial hot region. In addition, any structures and surfaces near the blast will influence the result. SPHC can easily handle all of these effects.

Actual blasts usually show a strong shock followed by a period of strong reverse flow, as seen in some of the early atomic testing in Nevada. This could be followed by secondary shocks. These effects are characteristic of actual ground blasts.

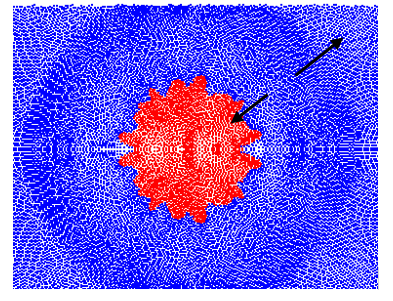
The test case shown here, labeled “blastJ” begins with an extended region near the origin containing hot gas characteristic of a PBX explosive described by a JWL equation of state. Geometry is 1D / spherical. Final plot for this run shows a double blast wave, but at later time the outer wave dominates, and the result tends toward the normal blast wave case discussed above. Run time for this test is 6.24 s.





Probe data for the density is shown above. Note the near constant velocity of the primary blast front and its decrease in strength – both caused by the relatively large initial radius of the blast.

The interface between the hot and cold initial gas region can be unstable if the velocity reverses in this region. This can be seen in two dimensional runs for this case, as seen in the snapshot below, colored on region, where the blast wave is at the outer edge of the frame (dark blue region), and the interface (red/blue) has developed a pronounced instability as it is driven inward. In field tests the red inner region is often seen as a smoke or dust cloud containing the remnants of the explosion.



Listing for the 1D case is shown below, file neme is “blastJ.inp”.

#====blast wave test case====

problem\_title "Blast 1D"

run\_title "JWL"

dimension = 1

nparticles = 1000

max\_time = 10e-6

restart\_dumps = 20

plot\_dumps 20

plot\_press

hist\_dumps = 800

err\_tol 0.01

dump\_accel

dump\_eos

spherical // spherical wave!

Outer = 4

Bdry = 0.2\*Outer

#---------boundaries-------

set\_boundary left

location = 0.

side low

end\_boundary

set\_boundary right

location = Outer

side high

end\_boundary

#-------data probes--------

set\_probe fixed 0.3\*Outer 0. 0.

set\_probe fixed 0.5\*Outer 0 0

set\_probe fixed 0.7\*Outer 0 0

#-------regions----------

Mbar = 1.e12

Mtrps = 100

set\_region "high temp"

material pg

eos jwl

set\_jwl 0.45 16.689\*Mbar 0.5969\*Mbar 5.9 2.1 // PBX9501

set\_jwl0 1.762 3500 0.0579\*Mbar 9000\*Mtrps

end\_region

set\_region "low temp"

material pg

eos pg

gamma = 1.6667

mu = 1.

density = 1.

temp = 300.

end\_region

#---- model build follows----

begin\_region "high temp"

part\_mult = Bdry/Outer

do\_sphere Bdry

begin\_region "low temp"

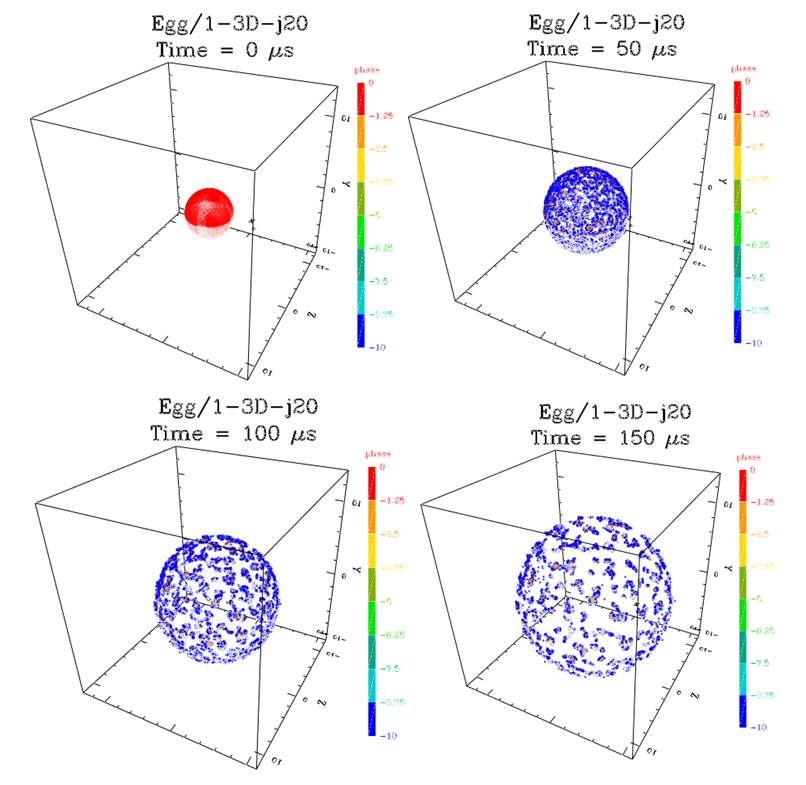
part\_mult = (Outer-Bdry)/Outer

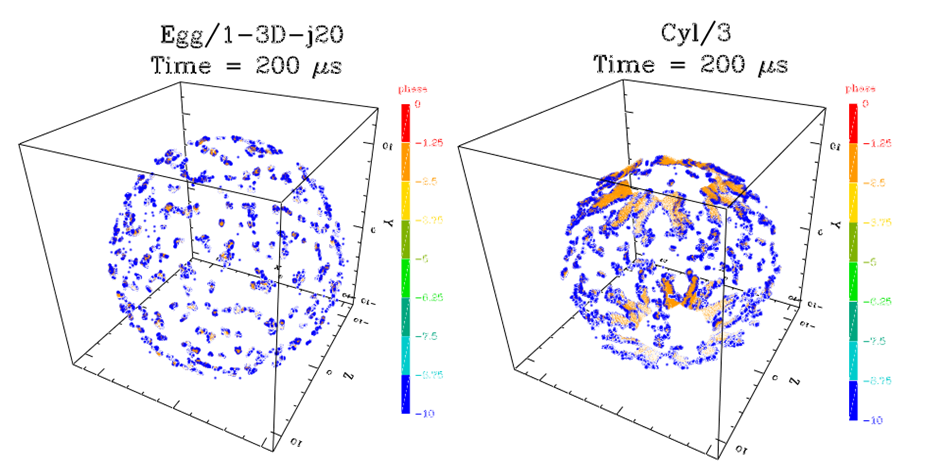
do\_sphere Outer Outer-Bdry

smooth 2. 1 1

### Debris Cloud

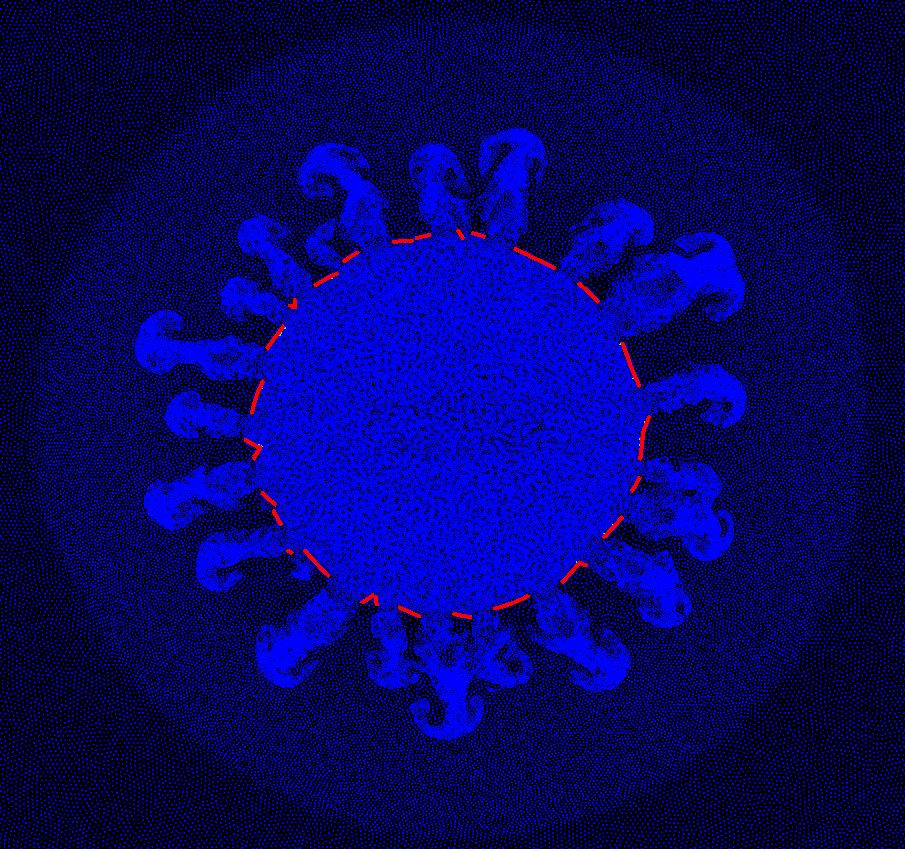
In addition to the expanding hot gas behind a blast wave, in most explosive incidents a cloud of solid debris particles is also formed. In this case the failure mechanism for the containment structure (fuel tank, etc.), is the primary mechanism for the debris cloud formation. The cloud expansion velocity will be determined by the interior hot gas pressure, as well as the details of the release of this pressure and the energy required to fracture the material. This test case models the formation and initial expansion of a debris cloud. A small aluminum sphere is modeled with an initial outward velocity. The velocity is large enough to overcome the fracture strength of the material (fracture model is default fracture with Weibull fault distribution). This test case can easily be modified to apply to a realistic structure, actual explosive and the effect of atmospheric deceleration of the debris at late times. In most cases, the debris cloud will expand with approximately constant radial velocity while the blast wave is decelerating as more gas is swept up, and the debris will eventually overtake the blast front. This could be important for some situations, and make the debris cloud the primary threat to an escape vehicle or other nearby structure. These plots are colored on material phase, starting with red (solid), moving through orange (plastic deformation), to blue (fractured). Run time for this model is 1.2 min.





The upper 4 frame sequence shows the expanding debris cloud from the hollow sphere. The lower two frames show the final configuration (left) as compared to the debris cloud produced by the expanding cylinder test case (Cyl/3, right) discussed in the standard case list, but run to the same late time as this test. This is probably more representative of an actual application minus the massive parts, such as engines, nosecones, etc., that would be traveling at lower velocities and thus located nearer the origin at late time.

As an example of a case involving all three components of an explosive failure, the figure below shows the very early stage of a tank failure, including ruptured steel tank debris (red, view is down the axis of a cylindrical tank)), expanding high temperature gas (light blue), mixing layer of this gas with an expernal atmosphere (blue mushrooms), and the blast front (medium blue, near outer edge). At later times the blast wave will expand and slow, the debris will accelerate and disperse.



Setup file for the “egg” test is shown below (file name = egg3D.inp)

#==== exploding egg test case====

problem\_title "Egg"

run\_title "1-3D-j20"

#---units---

Usec = 1.e-6

Msec = 1.e-3

Usec = 1.e-6

dimension = 3

nparticles = 5500

space\_adjust = 1.1

max\_time = 200\*Usec

restart\_dumps = 5

plot\_dumps = 20

plot\_v

hist\_dumps = 400

h\_inp = 1.0

h\_vary = true

error\_control false

pert\_size 20 // (up from 1)

symmetry x

symmetry y

symmetry z

#--------strength--------

strength\_model elas\_perf\_plas

fracture

#-------regions----------

set\_region ball

material al

eos grun

strength\_model elas\_perf\_plas

fracture

weibull

end\_region

#---- model build follows----

Vel = 5.e4

Radius = 3

Thick = 0.10

begin\_region ball

part\_mult = 1

do\_sphere Radius Thick

radial\_velocity

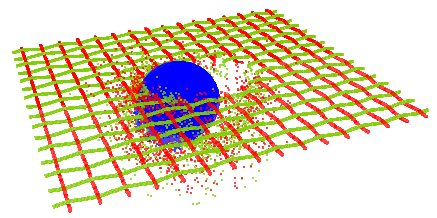
points 0 Vel 10 Vel

end\_velocity

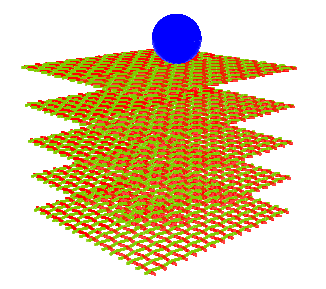
### Woven Structures

Many fabric and composite materials are now constructed of woven layers, sometimes with complex geometries. In this section we present a “subroutine” to generate a woven layer, and illustrate how, through repeated calls to the routine, more complex structures, consisting of several layers of woven material, can be generated.

A typical impact on a woven mesh might look like this:



Repeated calls in the setup to the mesh subroutine, could produce the following setup:



The spacing between the wires and the layers can be varied to produce a variety of materials. These models require some information about the wire material properties. Usually, each wire is strong along its length, and may stretch across its direction. The degree of attachment between wires can also be varied. Models have been successfully constructed in this way for fabrics, metal meshes, and composite materials.

Here is the routine for constructing two mesh layers. Note that in comments it shows how the two layers could be built separately, then repeats the same calls using a loop over layers with subscripted layer names.

/\*--- impact vulnerability simulation ------\*/

/\* (c) 11/15/2003 - Stellingwerf Consulting \*/

// woven target test case

// subroutine version

problem\_title "CASE M"

run\_title "3D"

#----run parameters----

Layers = 2

dimension = 3

nparticles = Layers\*30000

#---impactor parameters---

P\_rad = 0.25\*2.54/2

Vel = 7.0e5

Angle = 45.

#---mesh parameters---

T\_thick = 0.02 // 32 gauge wires

T\_width = 2.54/2 // 1/2 width in Z

T\_length = 2.54 // length in X

Nwires = 8 // in z direction/2

N2wires = 16 // in x direction

Mesh\_top = 0

/\* miscl \*/

symmetry z // build 1/2 problem

space\_adjust = 1.2

err\_tol = 1.e-3

debug\_part = 20

slip\_regions

/\* run control \*/

max\_time = 3.e-6;

restart\_dumps = 5

plot\_dumps = maxt/1.e-7

hist\_dumps = 100

#----derived quantities----

Theta = rad\*Angle

V1 = 1.333\*pi\*cub(P\_rad)/2;

Vx = pi\*sq(T\_thick/2)\*T\_length\*Nwires;

Vy = pi\*sq(T\_thick/2)\*T\_width\*N2wires

F1 = V1/10

Fx = Vx/2

Fy = Vy/2

Ft = F1+Layers\*(Fx+Fy)

#----strength----

strength\_model = high\_str\_rate

fracture

#----define regions----

set\_region ball

material al2017

eos grun

end\_region

set\_region meshx

material = al1350

eos grun

end\_region

set\_region meshy

material = al1350

eos grun

end\_region

#----begin model build----

/\* projectile \*/

begin\_region ball

part\_mult = F1/Ft

do\_sphere P\_rad P\_rad

translate\_reg P\_rad\*tan(Theta) P\_rad+T\_thick 0.

velocity\_reg -Vel\*sin(Theta) -Vel\*cos(Theta) 0.

#----call mesh build subroutine----

//begin\_item mesh1

//read\_data mesh\_sub.inp

//end\_item mesh1

// Mesh\_top = -.5

//begin\_item mesh2

//read\_data mesh\_sub.inp

//end\_item mesh2

begin\_loop\_i = 1 Layers

begin\_item mesh[ii]

Mesh\_top = -.5\*(ii-1)

read\_data mesh\_sub.inp

end\_item mesh[ii]

end\_loop\_i

In this setup, repeated calls to the file “mesh\_sub.inp” are executed. This subroutine for a general mesh layer is given below. Full instructions for its use are given in the comments at the top. Alternative commands for rectangular or round “wires” are shown in the listing.

/\* MESH\_SUB - generate mesh target as subroutine \*/

/\* Copyright (c) 2003 - Stellingwerf Consulting \*/

/\* this generates a half-mesh for a "symmetry z" run \*/

# DEFINE THESE REGIONS:

# meshx - X-direction wires

# meshy - Y direction wires

# Define these user variables:

# T\_thick = wire thickness in cm, .02 = 32 gauge

# T\_width = 1/2 width (Z)

# T\_length = full length (X)

# Nwires = # of wires in Z direction / 2

# N2wires = # of wires in X direction

# Fx, Fy, Ft - volume multipliers, see below

# Mesh\_top - Y position of the top of the mesh

# set "symmetry z" for rest of setup

# set "slip\_regions" to control stickiness of wires

# typical volume setup, depends on other regions

# V1 = 1.333\*pi\*cub(P\_rad)/2; // for sphere impactor

# Vx = pi\*sq(T\_thick/2)\*T\_length\*Nwires;

# Vy = pi\*sq(T\_thick/2)\*T\_width\*N2wires

# F1 = V1/10 // do adjustments to the zoning here

# Fx = Vx/2

# Fy = Vy/2

# Ft = F1+Fx+Fy // define Ft as the total vol multiplier

/\*--------------START---------------------------------------------------------\*/

/\* derived wire properties \*/

Wire\_len = T\_length/N2wires

Wire\_len2 = T\_width/Nwires

Wiggle = T\_thick/Wire\_len

/\* shield - X direction wires \*/

// turn off the z symmetry for the wire build

set\_no\_neg 0 0 0

part\_mult = Fx/(Ft\*Nwires\*N2wires)

begin\_region meshx

begin\_loop\_i 1 Nwires

Sgni = -1^ii

begin\_loop\_j 1 N2wires/2

Sgnj = -1^jj

// build wires from four skewed sections

//do\_trap Wire\_len/2. T\_thick T\_thick -Sgni\*Wiggle 0. // square wires

do\_cylinder T\_thick/2 Wire\_len/2. // round wires

rotate\_reg 0 0 90 // round

skew\_reg -Sgni\*Wiggle 0. // round

translate\_reg Wire\_len/4. -Sgni\*T\_thick/4. 0.

translate\_reg -Sgnj\*(2\*jj-1)\*Wire\_len/2 Sgni\*T\_thick/2. (ii-1)\*T\_width/Nwires

//do\_trap Wire\_len/2. T\_thick T\_thick -Sgni\*Wiggle 0.

do\_cylinder T\_thick/2 Wire\_len/2.

rotate\_reg 0 0 90

skew\_reg -Sgni\*Wiggle 0.

translate\_reg -Wire\_len/4. (-1^ii)\*T\_thick/4. 0.

translate\_reg Sgnj\*(2\*jj-1)\*Wire\_len/2 -Sgni\*T\_thick/2. (ii-1)\*T\_width/Nwires

//do\_trap Wire\_len/2. T\_thick T\_thick Sgni\*Wiggle 0.

do\_cylinder T\_thick/2 Wire\_len/2.

rotate\_reg 0 0 90

skew\_reg Sgni\*Wiggle 0.

translate\_reg -Wire\_len/4. -Sgni\*T\_thick/4. 0.

translate\_reg -Sgnj\*(2\*jj-1)\*Wire\_len/2 Sgni\*T\_thick/2. (ii-1)\*T\_width/Nwires

//do\_trap Wire\_len/2. T\_thick T\_thick Sgni\*Wiggle 0.

do\_cylinder T\_thick/2 Wire\_len/2.

rotate\_reg 0 0 90

skew\_reg Sgni\*Wiggle 0.

translate\_reg Wire\_len/4. Sgni\*T\_thick/4. 0.

translate\_reg Sgnj\*(2\*jj-1)\*Wire\_len/2 -Sgni\*T\_thick/2. (ii-1)\*T\_width/Nwires

end\_loop\_j

end\_loop\_i

merge\_sub\_regions

translate\_reg 0 0 Wire\_len2/2

#----position the mesh

translate\_reg 0 Mesh\_top 0

/\* shield - Y direction wires \*/

part\_mult = Fy/(Ft\*Nwires\*N2wires)

begin\_region meshy

begin\_loop\_i 1 N2wires

Sgni = -1^ii

begin\_loop\_j 1 Nwires/2

Sgnj = -1^jj

// build wires from four skewed sections

//do\_trap Wire\_len2/2. T\_thick T\_thick -Sgni\*Wiggle 0. // square wires

do\_cylinder T\_thick/2 Wire\_len2/2. // round wires

rotate\_reg 0 0 90 // round

skew\_reg -Sgni\*Wiggle 0. // round

translate\_reg Wire\_len2/4. -Sgni\*T\_thick/4. 0.

translate\_reg -Sgnj\*(2\*jj-1)\*Wire\_len2/2 Sgni\*T\_thick/2. (ii-1)\*T\_length/N2wires

//do\_trap Wire\_len2/2. T\_thick T\_thick -Sgni\*Wiggle 0.

do\_cylinder T\_thick/2 Wire\_len2/2.

rotate\_reg 0 0 90

skew\_reg -Sgni\*Wiggle 0.

translate\_reg -Wire\_len2/4. (-1^ii)\*T\_thick/4. 0.

translate\_reg Sgnj\*(2\*jj-1)\*Wire\_len2/2 -Sgni\*T\_thick/2. (ii-1)\*T\_length/N2wires

//do\_trap Wire\_len2/2. T\_thick T\_thick Sgni\*Wiggle 0.

do\_cylinder T\_thick/2 Wire\_len2/2.

rotate\_reg 0 0 90

skew\_reg Sgni\*Wiggle 0.

translate\_reg -Wire\_len2/4. -Sgni\*T\_thick/4. 0.

translate\_reg -Sgnj\*(2\*jj-1)\*Wire\_len2/2 Sgni\*T\_thick/2. (ii-1)\*T\_length/N2wires

//do\_trap Wire\_len2/2. T\_thick T\_thick Sgni\*Wiggle 0.

do\_cylinder T\_thick/2 Wire\_len2/2.

rotate\_reg 0 0 90

skew\_reg Sgni\*Wiggle 0.

translate\_reg Wire\_len2/4. Sgni\*T\_thick/4. 0.

translate\_reg Sgnj\*(2\*jj-1)\*Wire\_len2/2 -Sgni\*T\_thick/2. (ii-1)\*T\_length/N2wires

end\_loop\_j

end\_loop\_i

merge\_sub\_regions

translate\_reg 0 0 Wire\_len/2

#----reposition Y wires

translate\_reg 0 0 -T\_length/2

rotate\_reg 0 90 0

translate\_reg 0 0 T\_width/2

#----position the mesh

translate\_reg 0 Mesh\_top 0

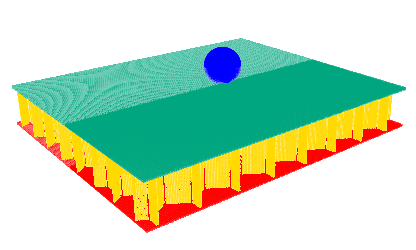
#----reset symmetry----

set\_no\_neg 0 0 1

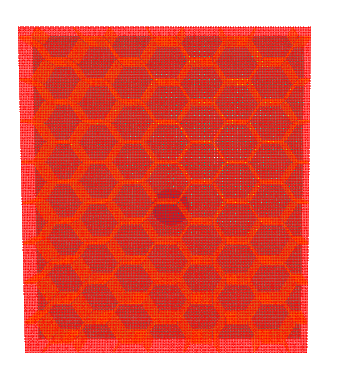
/\*--------------DONE!---------------------------------------------------------\*/

### Folded Structures

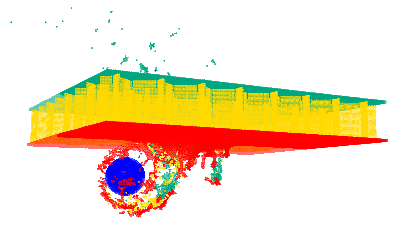
To conserve weight, spacecraft structures are often constructed of folded layers of material that contain many angled walls and much open space. These structures are challenging to model due to the many thin layers and surfaces, as well as complex geometry. SPHC can construct these cases using the various iterative capabilities provided in the setup machinery. One example will be given here – a hex-honeycomb structure consisting of two parallel plates with a honeycomb of perpendicular walls enclosed. The model is that of a high velocity sphere impacting such a structure. The setup, colored on region, looks like this:



Top view:



The final result following penetration looks like this:



The setup routine for this case, with some optional features included as comments, is shown here:

/\*--- HexC\_3D impact vulnerability simulation ------\*/

/\* (c) 11/20/2003 - Stellingwerf Consulting \*/

// hex honeycomb target test case

// subroutine version: calls hexc\_sub.inp

problem\_title "HexC"

run\_title "3D"

#----run parameters----

dimension = 3

nparticles = 200000

#---impactor parameters---

P\_rad = 0.25\*2.54/2

Vel = 1.0e5

Angle = 45.

#---mesh parameters---

F1\_thick = .05 // top face thickness

F2\_thick = .05 // bot face thickness

H\_length = .25\*2.54 // hex size, face to face

W\_height = 2.54/4 // cell height

W\_thick = 0.02 // wall thickness

Ncells = 4 // in z direction/2 (even #)

N2cells = 8 // in x direction

/\* miscl \*/z

symmetry z // build 1/2 problem

space\_adjust = 1.2

err\_tol = 1.e-3

debug\_part = 20

slip\_regions

/\* run control \*/

max\_time = 3.e-5;

restart\_dumps = 5

plot\_dumps = 50

hist\_dumps = 100

#----derived quantities----

Theta = rad\*Angle

T\_width = Ncells\*H\_length\*sqrt(3.)

T\_length = N2cells\*H\_length

V1 = 1.333\*pi\*cub(P\_rad)/2

Vf1 = T\_width\*T\_length\*F1\_thick/2

Vf2 = T\_width\*T\_length\*F2\_thick/2

Vw = 1.5\*sqrt(3)\*W\_height\*W\_thick\*H\_length\*N2cells\*Ncells

F1 = V1

Ff1 = Vf1

Ff2 = Vf2

Fw = Vw // wall parts smaller

Ft = F1+Ff1+Ff2+Fw

// for 2-layer test case

//Ft = F1+2\*(Ff1+Ff2)+1.5\*Fw

#----strength----

strength\_model = high\_str\_rate

fracture

#----define regions----

set\_region ball

material al2017

eos grun

end\_region

set\_region face1

material = al1350

eos grun

end\_region

set\_region walls

material = al1350

eos grun

slip 1 // weld to faces

end\_region

set\_region face2

material = al1350

eos grun

end\_region

#----begin model build----

/\* projectile \*/

begin\_region ball

part\_mult = F1/Ft

do\_sphere P\_rad P\_rad

translate\_reg W\_height\*tan(Theta)/2 P\_rad 0.

velocity\_reg -Vel\*sin(Theta) -Vel\*cos(Theta) 0.

#----call HC build subroutine----

begin\_item hc

read\_file hexc2\_sub.inp

end\_item hc

//---------------------------------

// example code for multiple layers

// increase npart & Ft accordingly

//W\_height = 0.25 // layer 2 is thinner

// begin\_item hc2

// read\_file hexc2\_sub.inp

// end\_item hc2

//translate\_item hc2 0 -1 0

//---------------------------------

// example code for filler material

//set\_region fill

// material = al1350

// eos grun

// slip 1

//end\_region

//begin\_region fill

//set\_npart 5000

//do\_block T\_length W\_height T\_width

//translate\_reg 0 -W\_height/2 0

//make\_room\_item hc

The construction procedure for repetitive structures consists of constructing a single “generator” object – in this case three walls of the hex layer joined to make a “Y” shape that, when duplicated, generates the entire honeycomb layer. For objects such as this one with plates at odd angles, be careful that all the surfaces match perfectly by using “drop\_box” to square off the ends, or, possibly “make\_room\_reg” to eliminate overlap.

For this case, the hexagon layer is generated by the following subroutine, called “hexc2\_sub.inp”. Note that, as a by-product of the usual manufacturing technique of constructing the hex layers from multiple foil layers that are welded along one edge of the cell, then expanded to form the grid, two faces of each cell are double thickness. This is the version modeled here.

/\* Hexc2\_sub - generate hex honeycomb target as subroutine \*/

/\* Copyright (c) 11/20/2003 - Stellingwerf Consulting \*/

/\* this generates a half-target for a "symmetry z" run \*/

/\* top of honeycomb is placed at Y=0 \*/

/\* single honeycomb layer generated \*/

/\* double thickness parallel wall version \*/

# DEFINE THESE REGIONS:

# face1 - material for upper face sheet

# face2 - material for lower face sheet

# walls - material for wall separators

# set "slip 1" for walls to weld to face plates

# Define these user variables:

# F1\_thick = top face sheet thickness

# F2\_thick = bottom face sheet thickness

# H\_length = size of a hex cell, face to face (X)

# W\_height = distance (Y) between face sheets in cm

# W\_thick = wall thickness (Z) - thin wall

# Ncells = # of cells in Z direction / 2 (even)

# N2cells = # of cells in X direction

# Ff1, Ff2, Fw, Ft - volume multipliers, see below

# set "symmetry z" for rest of setup

# insert the command "get\_reg\_density walls W\_den" after the region defs

#--------typical volume setup, depends on other regions--------

# T\_width = Ncells\*H\_length\*sqrt(3.) // Z

# T\_length = N2cells\*H\_length // X

# V1 = 1.333\*pi\*cub(P\_rad)/2; // for sphere impactor

# Vf1 = T\_width\*T\_length\*F1\_thick/2; // each facesheet

# Vf2 = T\_width\*T\_length\*F2\_thick/2; // each facesheet

# Vw = 1.5\*sqrt(3)\*W\_height\*W\_thick\*H\_length\*N2cells\*Ncells // cells

# F1 = V1 // do adjustments to the zoning here

# Ff1 = Vf1

# Ff2 = Vf2

# Fw = 4\*Vw // make particles smaller in walls

# Ft = F1+Ff1+Ff2+Fw // define Ft as the total vol multiplier

/\*--------------START---------------------------------------------------------\*/

/\* local variables \*/

Wall\_len$ = H\_length/sqrt(3.)

get\_reg\_density walls W\_den$

get\_reg\_density face1 F1\_den$

get\_reg\_density face2 F2\_den$

#----do top face sheet----

begin\_region face1

part\_mult Ff1/Ft

do\_block T\_length F1\_thick T\_width

mass\_reg F1\_den$\*T\_length\*F1\_thick\*T\_width/2

translate\_reg 0 -F1\_thick/2 0

#----turn off the z symmetry for the wall build----

set\_no\_neg 0 0 0

begin\_region walls

// basic tri-wall unit, taper wall ends to fit

begin\_item h\_unit

part\_mult = 2\*Fw/(6\*Ft\*Ncells\*N2cells)

do\_block Wall\_len$/2 W\_height 2\*W\_thick

// adjust the mass of the initial element

mass\_reg 2\*W\_den$\*(Wall\_len$/2)\*W\_height\*W\_thick

translate\_reg Wall\_len$/4 0 0

rotate\_reg 0 60 0

delete\_box -H\_length H\_length -W\_height W\_height -H\_length 0.

rotate\_reg 0 60 0

delete\_box -H\_length H\_length -W\_height W\_height -H\_length 0.

rotate\_reg 0 150 0

part\_mult = Fw/(6\*Ft\*Ncells\*N2cells)

do\_block Wall\_len$/2 W\_height W\_thick

mass\_reg W\_den$\*(Wall\_len$/2)\*W\_height\*W\_thick

translate\_reg Wall\_len$/4 0 -W\_thick/2

rotate\_reg 0 60+atan(W\_thick/(Wall\_len$))/rad 0

delete\_box -H\_length H\_length -W\_height W\_height -H\_length 0.

rotate\_reg 0 -30 0

dup\_reg

reflect\_reg 1 0 0

merge\_sub\_regions

end\_item h\_unit

// now build all the hexes

begin\_loop\_i = 1 Ncells/2

begin\_loop\_j = 1 N2cells

if ii\*jj=1

// skip first hunit-already done above

else

dup\_item h\_unit

translate\_reg (jj-1)\*H\_length 0 (ii-1)\*3\*Wall\_len$

end\_if

dup\_item h\_unit

rotate\_reg 0 180 0

translate\_reg (jj-1)\*H\_length 0 (ii-1)\*3\*Wall\_len$+2\*Wall\_len$

dup\_item h\_unit

translate\_reg (jj-1)\*H\_length+H\_length/2 0 3\*(ii-1)\*Wall\_len$+1.5\*Wall\_len$

dup\_item h\_unit

rotate\_reg 0 180 0

translate\_reg (jj-1)\*H\_length+H\_length/2 0 3\*(ii-1)\*Wall\_len$+0.5\*Wall\_len$

end\_loop\_j

end\_loop\_i

#----delist temporary building block----

delist\_item h\_unit

#----finish up----

merge\_sub\_regions

translate\_reg -N2cells\*H\_length/2 -W\_height/2 Wall\_len$/2

#----position the mesh----

translate\_reg H\_length/4 -F1\_thick 0

#----reset symmetry----

set\_no\_neg 0 0 1

#----do bottom face sheet----

begin\_region face2

part\_mult Ff2/Ft

do\_block T\_length F2\_thick T\_width

mass\_reg F2\_den$\*T\_length\*F2\_thick\*T\_width/2

translate\_reg 0 -W\_height-F2\_thick/2-F1\_thick 0

## References

Abadi, M. G., Lambas, D. G., Tissera, P. B. (1996) Cosmological Simulations with Smoothed Particle Hydrodynamics, Symposium, Volume 168, 577. <http://adsabs.harvard.edu/full/1996IAUS..168..577A>

Amdahl, D. L. (1993) Modelling Aerodynamic Problems Using Smoothed Particle Hydrodynamics (SPH), Transactions of the Society of Automotive Engineers, 102, 1, 1743-1750. <https://www.jstor.org/stable/44740127?seq=1>

Balsara, Dinshaw S. (1995) von Neumann Stability Analysis of Smoothed Particle Hydrodynamics -- Suggestions for Optimal Algorithms, Journal of Computational Physics, 121, 2, 357. <https://www.sciencedirect.com/science/article/pii/S002199919590221X>

Benz, W. (1988) Applications of Smoothed Particle Hydrodynamics (SPH) to Astrophysical Problems, Computer Physics Communications, 48, 97-105. <https://www.sciencedirect.com/science/article/pii/0010465588900276>

Benz, W. (1989) Smooth Particle Hydrodynamics: A Review. Harvard-Smithsonian Center for Astrophysics Preprint No. 2884. <https://link.springer.com/chapter/10.1007/978-94-009-0519-1_16>

Benz, W. and Asphaug, E. (1995) Simulations of Brittle Solids using Smooth Particle Hydrodynamics, Computer Physics Communications, 87, 253-265. <https://www.sciencedirect.com/science/article/pii/0010465594001763>

Benz, W. (1990). Smooth Particle Hydrodynamics: A Review. J. R. Buchler (ed.), The Numerical Modeling of Nonlinear Stellar Pulsations, 269-288. <https://link.springer.com/chapter/10.1007/978-94-009-0519-1_16>

Hans A. Bethe, Klaus Fuchs, Joseph O. Hirschfelder, John L. Magee, Rudolph E. Peierls, John von Neumann, (1947) Blast Wave, Los Alamos Report LA2000. <https://apps.dtic.mil/sti/pdfs/ADA384954.pdf>

Bicknell, G. V. (1991) The Equations of Motion of Particles in Smoothed Particle Hydrodynamics, SIAM Journal on Scientific and Statistical Computing, 12, 5, 1198. <https://epubs.siam.org/doi/abs/10.1137/0912064>

Boffin, H. M. J. and Anzer, U. (1994) Numerical studies of wind accretion using SPH. 1. 2D simulations, Astronomy and astrophysics, 284, 3, 1026.

<http://adsabs.harvard.edu/full/1994A%26A...284.1026B>

Bravo, E. and Garcia-Senz, D. (1995) Smooth Particle Hydrodynamics Simulations of Deflagrations in Supernova, The Astrophysical Journal, 450, L17-L21. <https://iopscience.iop.org/article/10.1086/309654/meta>

Chakrabarti, Sandip K. and Molteni, Diego, (1993) Smoothed Particle Hydrodynamics Confronts Theory: Formation Of Standing Shocks In Accretion Disks And Wings Around Black Holes, The Astrophysical journal, 417, 2, 671. <https://ui.adsabs.harvard.edu/abs/1993ApJ...417..671C/abstract>

Couchman, H. M. P., Thomas, P.A. and Pearce, F.R., (1995) Hydra: An Adaptive-Mesh Implementation of P3M-SPH, The Astrophysical Journal, 452, 797-813. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.56.2831&rep=rep1&type=pdf>

Davies, M. B., Ruffert, M., Benz, W. and Muller, E. (1993) A Comparison Between SPH and PPM: Simulations of Stellar Collisions, Astronomy and Astrophysics, 272, 2, 430-441. <http://adsabs.harvard.edu/full/1993A%26A...272..430D>

Dyka, C T and Ingel, R P (1995) An approach for tension instability in smoothed particle hydrodynamics (SPH), Computers & structures, 57, 4, 573. <https://www.sciencedirect.com/science/article/abs/pii/004579499500059P>

Evrard, August E. (1988) Beyond N-Body: 3D cosmological gas dynamics, MNRAS, 235, 911. <https://ui.adsabs.harvard.edu/abs/1988MNRAS.235..911E/abstract>

Flebbe, O.; Muenzel, S.; Herold, H.; Riffert, H. (1994) Smoothed Particle Hydrodynamics: Physical Viscosity and the Simulation of Accretion Disks, The Astrophysical Journal, 431, 754. <http://adsabs.harvard.edu/full/1994ApJ...431..754F>

Fulbright, M. S., Benz, W. and Davies, M. B. (1995) A Method of Smoothed Particle Hydrodynamics Using Spheroidal Kernels, The Astrophysical Journal, 440, 1, 254. <http://adsabs.harvard.edu/full/1995ApJ...440..254F>

Fulk, D.A. (1994), A Numerical Analysis of Smoothed Particle Hydrodynamics, PhD. Thesis, Air Force Institute of Technology.

Fulk, D. A. and Quinn, D. W. (1995) Hybrid formulations of smoothed particle hydrodynamics, International journal of impact engineering., 17, 1/3, p 329-340. <https://www.sciencedirect.com/science/article/abs/pii/0734743X9599859P>

Fulk, David A and Quinn, Dennis W, (1996). An Analysis of 1-D Smoothed Particle Hydrodynamics Kernel, Journal of computational physics, Volume 126, Number 1, pp. 165

Gingold R. A. and J. J. Monaghan (1982) Kernel Estimates as a Basis for General Particle Methods in Hydrodynamics. Journal of Computational Physics, 46, 429-453. <https://www.sciencedirect.com/science/article/pii/0021999182900250>

Gingold R. A. and J. J. Monaghan (1977) Smoothed particle hydrodynamics: Theory and application to non spherical stars. Mon. Not. Roy. Astron. Soc., 181, 375-389. <https://academic.oup.com/mnras/article/181/3/375/988212>

Grossman, Scott A. and Narayan, Ramesh (1993) A Theory Of Nonlocal Mixing-Length Convection. II. Generalized Smoothed Particle Hydrodynamics Simulations, The Astrophysical journal. Supplement series., 89, 2, 361. <http://adsabs.harvard.edu/full/1993ApJS...89..361G>

Henneken, E. A. C. and Icke, V. (1993) SPH faces Emery's Jump, Computer physics communications, 74, 2, 239-246. <https://www.sciencedirect.com/science/article/pii/001046559390094S>

Hernquist, Lars, (1993) Some Cautionary Remarks About Smoothed Particle Hydrodynamics, The Astrophysical journal, 404, 2, 717. <http://adsabs.harvard.edu/full/1993ApJ...404..717H>

Hernquist, L. and N. Katz, (1989) TreeSPH: A unification of SPH with the hierarchical tree method. Astrophysical Journal Supplement, 70, 419-446. <http://adsabs.harvard.edu/full/1989ApJS...70..419H>

Hiotelis, N.; Voglis, N. (1991) Smooth particle hydrodynamics with locally readjustable resolution in the collapse of a gaseous protogalaxy, Astronomy and astrophysics, 243, 2, 333. <https://www.researchgate.net/publication/234272010_Smooth_particle_hydrodynamics_with_locally_readjustable_resolution_in_the_collapse_of_a_gaseous_protogalaxy>

Johnson, G. R. (1996) Artificial viscosity effects for SPH impact computations, International journal of impact engineering, 18, 5, 477. <https://www.sciencedirect.com/science/article/abs/pii/0734743X9500051B>

Johnson, G. R.; Beissel, S. R. (1996) Normalized Smoothing Functions for SPH Impact Computations, International journal for numerical methods in engineering, 39, 16, 2725. <https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0207%2819960830%2939%3A16%3C2725%3A%3AAID-NME973%3E3.0.CO%3B2-9>

Johnson, G R; Stryk, R A; Beissel, S R (1996) SPH for high velocity impact computations, Computer methods in applied mechanics and engineering, 139, 1-4, 347. <https://www.sciencedirect.com/science/article/pii/S0045782596010894>

Kang, Hyesung , Jeremiah P. Ostriker, Renyue Cen, Dongsu Ryu, Lars Hernquist, August E Evrard, Greg L. Bryan, Michael L.Norman (1994) A Comparison of Cosmological Hydrodynamical Codes, Astrophysical Journal, 430, 83. <https://arxiv.org/abs/astro-ph/9404014>

Laguna, Pablo; Miller, Warner A.; Zurek, Wojciech H. (1993). Smoothed Particle Hydrodynamics Near A Black Hole, The Astrophysical journal, Volume 404, Number 2, p 678. <http://adsabs.harvard.edu/full/1993ApJ...404..678L>

Lanzafame, G.; Belvedere, G.; Molteni, D. (1993). A three-dimensional smoothed particle hydrodynamics simulation of the active phase of SS Cyg-type discs and its implications for the mass transfer burst model, Monthly notices of the Royal Astronomical Society, Volume 263, Number 4, p 839. <https://academic.oup.com/mnras/article/263/4/839/983285>

Lancaster, P. and K. Salkauskas (1980) Surfaces Generated by Moving Least Squares Methods, Mathematics of Computation, 37, 155, 141-158. <https://www.ams.org/journals/mcom/1981-37-155/S0025-5718-1981-0616367-1/>

Lattanzio, J.C., J.J. Monaghan, H. Pongracic and M.P. Schwartz (1986) Controlling Penetration, SIAM Journal on Scientific and Statistical Computing, 7, 2, 591-598. <https://epubs.siam.org/doi/abs/10.1137/0907039?journalCode=sijcd4>

Libersky, L.D., Petschek, A. G., Carney, T.C., Hipp, J. R., and Allahdadi, F. A. (1993) High Strain Lagrangian Hydrodynamics, A Three Dimensional SPH Code for Dynamic Material Response, Journal of Computational Physics, 109, 67-75. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1022.2626&rep=rep1&type=pdf>

Liu, W. K., S. Jun, D. T. Sihling, Y. Chen and W. Hao (1997) Multiresolution Reproducing Kernel Particle Method for Computational Fluid Dynamics , International Journal of Numerical Method in Fluids, 24,1391-1415. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.55.7701&rep=rep1&type=pdf>

Lorimer, G.S. (1986) The Kernel Method of Air Quality Modeling - I. Mathematical Foundation, Atmospheric Environment, 20, 7, 1447-1452. <https://www.sciencedirect.com/science/article/abs/pii/0004698186900168>

Lubow, S.H. , J.E. Pringle, R.R. Kerswell (1993) Tidal Instability of Accretion Disks, ApJ, 419, 758. <http://adsabs.harvard.edu/full/1993ApJ...419..758L>

Lucy L. (1977). A Numerical Approach to Testing the Fission Hypothesis, Astronomical Journal, 82, 1013-1024 . <http://adsabs.harvard.edu/full/1977AJ.....82.1013L7>

Mandell, David A. and Charles A. Wingate (1994) Prediction of Material Strength and Fracture of Glass Using the SPHINX Smooth Particle Hydrodynamics Code, Los Alamos National Laboratory Report, LA-12830.

<https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-12830>

Mandell, David A., Charles A. Wingate and Robert F. Stellingwerf (1995) Prediction of Material Strength and Fracture of Brittle Materials Using the SPHINX Smooth Particle Hydrodynamics Code, Engineering Mechanics, Proceedings of 10th Conference, May 21-24, 1995. <https://www.osti.gov/servlets/purl/10103323>

Mandell, D A., C. A. Wingate and L. A. Schwalbe (1996) Simulation of a Ceramic Impact Experiment Using the SPHINX Smooth Particle Hydrodynamics Code, 16th International Symposium on Ballistics, San Francisco, CA, 23-28 September 1996, Los Alamos National Lab Report LA-UR-96-1878.

<https://www.osti.gov/servlets/purl/266913>

Mandell, D.A., Wingate, C.A., Schwalbe, L.A (1996) Computational brittle fracture using smooth hydrodynamics, LA-UR-96-2840. <https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-96-2840>

Mann, Patrick J. (1993) Smoothed Particle Hydrodynamics Applied to Relativistic Spherical Collapse, Jour. Comp. Phys., 107, 188. <https://www.sciencedirect.com/science/article/pii/S0021999183711356>

Martin, T. J.; Davey, S. C. (1995) Application of smoothed particle hydrodynamics to atmospheric motions in interacting binary systems, Monthly notices of the Royal Astronomical Society, Volume 275, Number 1, p 31. <https://ui.adsabs.harvard.edu/abs/1995MNRAS.275...31M/abstract>

McCormick, C. S.; Miller, T. F., (1994). Simulation of Shock-Wave Reflection in a One-Dimensional Shock Tube Using Smoothed Particle Hydrodynamics, Numerical heat transfer. Part B, Fundamentals. Volume 26, Number 4 page 473. <https://pennstate.pure.elsevier.com/en/publications/simulation-of-shock-wave-reflection-in-a-one-dimensional-shock-tu>

Meglicki, Z., (1994) Verification and accuracy of smoothed particle magnetohydrodynamics, Computer physics communications, Volume 81, Number 1/2 , p 91. <https://www.sciencedirect.com/science/article/pii/0010465594901139>

Molteni, D.; Gerardi, G.; Chakrabarti, S. K. (1994). Simulation of Interactions of an Orbiting Compact Star with an Accretion Disk by Smoothed Particle Hydrodynamics, The Astrophysical journal, Volume 436, Number 1 Number 1, p 249. <http://adsabs.harvard.edu/full/1994ApJ...436..249M>

Molteni, Diego; Lanzafame, Giuseppe; Chakrabarti, Sandip K. (1994). Simulations Of Thick Accretion Disks With Standing Shocks By Smoothed Particle Hydrodynamics, The Astrophysical journal, Volume 425, Number 1, p 161. <http://adsabs.harvard.edu/full/1994ApJ...425..161M>

Monaghan, J. J. and J. A. Morris (1993). SPH Masters Negative Stress. In the proceedings of the "Workshop on Advances in Smooth Particle Hydrodynamics", Los Alamos National Laboratory Report, LA-UR-93-4375.

Monaghan, J. J. (1982) Why Particle Methods Work, , SIAM Journal on Scientific and Statistical Computing, 3, 422-433. <https://epubs.siam.org/doi/abs/10.1137/0903027>

Monaghan, J. J. (1988) An Introduction to SPH, Computer Physics Communications, 48, 89-96.

Monaghan, J. J. (1989) On the Problem of Penetration in Particle Methods, Journal of Computational Physics, 82, 1-15. <https://dl.acm.org/doi/abs/10.5555/1718329.1718332>

Monaghan, J. J. (1992) Smoothed Particle Hydrodynamics, Annual Review of Astronomy and Astrophysics, 30, 543-574. <https://www.annualreviews.org/doi/abs/10.1146/annurev.aa.30.090192.002551?journalCode=astro>

Monaghan, J. J. (1994) Simulating Free Surface Flows with SPH, Journal of Computational Physics, 110, 399-406. <https://www.sciencedirect.com/science/article/pii/S0021999184710345>

Monaghan, J J (1996) Gravity currents and solitary waves, Physica D. Nonlinear phenomena, 98, Number 2-4, 523-533. <https://www.sciencedirect.com/science/article/abs/pii/0167278996001108>

Monaghan, J. J., Bicknell, P. J. and Humble, R. J. (1994) Volcanoes, Tsunamis and the demise of the Minoans, Physica D. Nonlinear phenomena, 77, Number 1/3, 217-228. <https://www.sciencedirect.com/science/article/abs/pii/016727899490135X>

Monaghan, J. J. and R. A. Gingold (1983) Shock Simulation by the Particle Method SPH, Journal of Computational Physics 52, 374-389. <https://www.sciencedirect.com/science/article/pii/0021999183900360>

Monaghan, J J and A. Kocharyan (1995) SPH simulation of multi-phase flow, Computer physics communications, 87, Number 1-2, 225. <https://www.sciencedirect.com/science/article/pii/001046559400174Z>

Morris, J.P. (1994) A Study of the Stability Properties of Smooth Particle Hydrodynamics, <http://www.maths.monash.edu.au/~jpm/work/short.ps>

<http://adsabs.harvard.edu/full/1996PASA...13...97M>

Nagasawa, M. and Kuwahara, K. (1993) Smoothed Particle Simulations of the Pyroclastic Flow, International journal of modern physics. B, Condensed matter physics, statistical physics, applied physics, 7, Number 9-10, 1979-1995. <https://www.worldscientific.com/doi/abs/10.1142/S0217979293002729>

Nelson, R. P. and Papaloizou, J. C. B., (1994) Variable smoothing lengths and energy conservation in smoothed particle hydrodynamics, Monthly notices of the Royal Astronomical Society, 270, 1, 1. <https://www.researchgate.net/publication/1809310_Variable_Smoothing_Lengths_and_Energy_Conservation_in_Smoothed_Particle_Hydrodynamics>

Noh, W.F. (1978) Errors for Calculations of Strong Shocks Using an Artificial Viscosity and an Artificial Heat Flux, Journal of Computational Physics, 72, 78-120. <https://www.sciencedirect.com/science/article/pii/002199918790074X>

Norris , Peter Martyn (1996) Radiatively-Driven Convection in Marine Stratocumulus Clouds, PhD. Thesis, University of California, San Diego.

Owen, J.M., Villumsen, J., Shapiro, P.R. and Martel, H. (1995) Adaptive smoothed particle hydrodynamics - Methodology II, LANL Preprint 9512078. <https://arxiv.org/abs/astro-ph/9512078>

Petschek, A.G. and Libersky, L. D. (1993) Cylindrical Smoothed Particle Hydrodynamics, Journal of Computational Physics, 109, 76-83. <https://www.sciencedirect.com/science/article/pii/S0021999183712003>

Randles, P W and Libersky, L D, (1996) Smoothed Particle Hydrodynamics: Some recent improvements and applications, Computer methods in applied mechanics and engineering, 139, 1-4, 375. <https://www.sciencedirect.com/science/article/pii/S0045782596010900>

Randles, P. W.; Carney, T. C.; Libersky, L. D.; Renick, J. D. (1995). Calculation of oblique impact and fracture of tungsten cubes using smoothed particle hydrodynamics, International journal of impact engineering, Volume 17, Number 4/6, p 661. <https://www.sciencedirect.com/science/article/abs/pii/0734743X9599889Y>

Rhoades, C. E. (1992) A fast algorithm for calculating particle interactions in smooth particle hydrodynamic simulations, Computer physics communications, 70, 3, 478-482. <https://www.sciencedirect.com/science/article/pii/001046559290109C>

Riffert, H., H. Herold, O. Flebbe and H. Ruder (1995). Numerical Aspects of the Smoothed Particle Hydrodynamics Method for Simulating Accretion Disks, Computer physics communications, 89, 1-16. <https://ui.adsabs.harvard.edu/abs/1995CoPhC..89....1R/abstract>

Schüssler, M. and D. Schmitt (1981) Comments on Smoothed Particle Hydrodynamics, Astronomy and Astrophysics, 97, 373. <http://adsabs.harvard.edu/full/1981A%26A....97..373S>

Serna, A., Alimi, J.-M. and Chieze, J.-P. (1996). Adaptive Smooth Particle Hydrodynamics and Particle-Particle Coupled Codes: Energy and Entropy Conservation, The Astrophysical Journal, 461, 884-896. <http://adsabs.harvard.edu/full/1996ApJ...461..884S>

Shapiro, Paul R, Martel, Hugo, Villlumsen, Jens V and Owen, J Michael, (1996). Adaptive Smoothed Particle Hydrodynamics, With Application To Cosmology Methodology, The Astrophysical journal. Supplement series., 103, 2, 269. <https://ui.adsabs.harvard.edu/abs/1996ApJS..103..269S/abstract>

Simpson, James C (1995). Numerical techniques for three-dimensional smoothed particle hydrodynamics simulations: applications to accretion disks, The Astrophysical journal, 448, 822-831. <http://adsabs.harvard.edu/full/1995ApJ...448..822S>

Smith, S. C., Houser, J. L. and Centrella, J. M. (1996) Simulations of Nonaxisymmetric Instability in a Rotating Star: A Comparison Between Eulerian and Smooth Particle Hydrodynamics, The Astrophysical Journal, 458, 236-256. <http://adsabs.harvard.edu/full/1996ApJ...458..236S>

Steinmetz, M.; Muller, E., (1993) On the capabilities and limits of smoothed particle hydrodynamics, Astronomy and astrophysics, 268, 1, 391. <http://adsabs.harvard.edu/full/1993A%26A...268..391S>

Stellingwerf, R. F. (1990) Smooth Particle Hydrodynamics, Advances in the Free-Lagrange Method, (Trease, Fritts, and Crowley, eds.), Springer Verlag, 239-247. <https://www.springer.com/gp/book/9783662138076>

Stellingwerf, R. F. (1989) SPHC Manual, Mission Research Corporation report MRC/ABQ-R-1237.

<https://www.stellingwerf.com/rfs-bin/index.cgi?action=GetDoc&id=2&filenum=1>

Stellingwerf, R. F. and C. A. Wingate (1992) Impact Modeling with Smooth Particle Hydrodynamics, : International journal of impact engineering, Volume 14, Number 1-4, 707-718. <https://www.sciencedirect.com/science/article/abs/pii/0734743X9390065F>

Swegle, J. W., Hicks, D. L. and Attaway, S. W. (1995) Smoothed Particle Hydrodynamics Stability Analysis, Journal of computational physics, 116, 1, 123-134. <https://www.sciencedirect.com/science/article/pii/S0021999185710108>

Swegle, J W and Attaway, S W, (1996) On the feasibility of using smoothed particle Hydrodynamics for underwater explosion calculations, Computational mechanics, 17, 3, 151-168. <https://ui.adsabs.harvard.edu/abs/1995CompM..17..151S/abstract>

Takeda, Hidenori, Miyama, Shoken M. and Sekiya, Minoru, (1994) Numerical Simulation of Viscous Flow by Smoothed Particle Hydrodynamics, Progress of theoretical physics, 92, 5, 939. <https://academic.oup.com/ptp/article/92/5/939/1868349>

Theuns, T. (1990) A Combination of SPH and N-Body 2 for Gas Dynamics in <http://adsabs.harvard.edu/full/1990Ap%26SS.170..221T> Star Clusters, Astrophysics and space science, Volume 170, Numbers 1-2, 221.

Umemura, Masayuki, Toshiyuki Fukushige, Junichiro Makino, Toshikazu Ebisuzaki, Daiichiro Sugimoto, Edwin L. Turner and Abraham Loeb (1993) Smoothed Particle Hydrodynamics with GRAPE-1A., Publ. Astron. Soc. Japan, 45, 311.

Watkins, S J, Bhattal, A S, Francis, N, Turner, J A, Whitworth, A P, (1996) A new prescription for viscosity in smoothed particle hydrodynamics, Astronomy & astrophysics Supplement series, 119, 1, 177. <https://aas.aanda.org/articles/aas/abs/1996/13/ds4605/ds4605.html>

Whitworth, A P, Bhattal, A S, Turner, J A and Watkins, S J, (1995) Estimating density in smoothed particle hydrodynamics, Astronomy and astrophysics, 301, 3, 929. <http://adsabs.harvard.edu/full/1995A%26A...301..929W>

Williams, M. and T. Yamada, (1990) A microcomputer-based forecasting model: potential applications for emergency response plans and air quality studies, J. Air Waste Manage. Assoc., 40, 1266-1274. <https://www.tandfonline.com/doi/abs/10.1080/10473289.1990.10466781>

Wingate, C.A. (1996) Modeling Partially Coupled Objects with Smooth Particle Hydrodynamics, Los Alamos National Laboratory Report LA-UR-96-3131. <https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-96-3131>

Wingate, C. A. and R. F. Stellingwerf (1993) Smooth Particle Hydrodynamics - The SPHINX and SPHC Codes. Los Alamos National Laboratory report, LA-UR-93-1938.

<https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-93-1938>

Wingate, C. A., R. F. Stellingwerf, R. F. Davidson and M. W. Burkett (1992) M <https://www.sciencedirect.com/science/article/abs/pii/0734743X9390075I>odels of High Velocity Impact Phenomena, Int. J. Impact Engng. 14, 819-830.

Yamada, T. and S. Bunker (1988) Development of a Nested Grid, Second Moment Turbulence Closure Model and Application to the 1982 ASCOT Brush Creek Data Simulation, Journal of Applied Meteorology, 27, 562-578. <https://journals.ametsoc.org/jamc/article/27/5/562/14419/Development-of-a-Nested-Grid-Second-Moment>

## End