REFERENCES

[1] Graham E W, Rodriguez A M, 1952, The characteristics of fuel motion which affect airplane dynamics. *Journal of Applied Mechanics.* 19 (1952) 381-8.

[2] Graham E W, 1951. The forces produced by fuel oscillation in a rectangular tank. *Douglas Aircraft Company Report*. SM-13748.

[3] Abramson H N, 1966, The dynamic behavior of liquids in moving containers, *NASA* SP-106.

[4] Fernando Meseguer-Garrido, 2003, On the sloshing of liquids in parallelepiped-shaped containers. *European Journal of Physics*. 24 (2003) 277-288.

[5] Vincenzo Armenio, Michele La Rocca, 1996, On the analysis of Sloshing of Water in rectangular containers: Numerical study and experimental validation. *Ocean Engineering.* 23 (1996) 705-739.

[6] Alain Cariou, Guido Casella, 1999, Liquid sloshing in ship tanks: A comparative study of numerical simulation. *Marine Structures*. 12 (1999) 183-198.

[7] Bass R L, Bowles E B, Cox P A, 1980, Liquid dynamic loads in LNG cargo tanks. *SNAME Trans*. 88 (1980) 103-126.

[8] Yamamoto S, Kataoka F, Shioda S, Ashitani Y, 1995, Study of impact due to sloshing in midsized LNG carrier. *International Journal Offshore Polar Engineering.* 5 (1995) 10-16.

[9] University of Washington, School of Oceanography. *Linear wave theory lecture notes.* Seattle, USA.

University of Pretoria etd – Kingsley, T C (2005)

[10] Van Dyke, 1982. The album of fluid motion. *Stanford*, California, USA.

[11] www.nhtsa.dot.gov, web page for National Highway Traffic Safety Administration. *US department of Transport*.

[12] www.mscsoftware.com, web page for *MSC-Software*, CA, USA.

[13] Zhang J, 2003, Non-linear wave theory course notes. ceprofs.tamu.edu/jzhang/ *Texas A&M University*. Texas, USA.

[14] Warnitchai P, Pinhaew T, 1998, Modelling of liquid sloshing in rectangular tanks with flow damping devices, *Engineering Structures*, 20 593-600.

[15] Modi V J, Munshi S R, 1998, An efficient liquid sloshing damper for vibration control, *Journal of Fluids and Structures*, 12 (1998) 1055-1071.

[16] Schotte J S, Ohayon R, 2003, Effect of gravity on a free-free elastic tank partially filled with incompressible liquid, *Journal of Fluids and Structures*, 18 (2003) 215-226.

[17] Faltinsen OM, 1978, A numerical non-linear method of sloshing in tanks with two dimentional flow, *Journal of Ship Research*, 18 (4) 224-241.

[18] Frandsen J B, 2003, Simulation of sloshing motion in fixed and vertically excited containers using a 2-D inviscid σ-transformed finite difference solver, *Journal of Fluids and Structures*, 18 (2003) 197-214.

[19] Fluent.Inc, 2002 , www.fluent.com, *fluent v.6 user manual*, Lebanon, NH.

[20] Wilcox DC, 1998, Turbulence modelling for CFD, *DCW Industries Inc*., La Canada, California.

[21] Hirt C W, Nichols B D, 1981, Volume of Fluid (VOF) method for the dynamics of free boundaries, *Journal of Computational Physics*. 39, 201-225.

University of Pretoria etd – Kingsley, T C (2005)

[22] Thomas B G, Zhang L, 2001, Mathematical Modelling of Fluid Flow in Continuous Casting, *ISIJ international*, vol. 41, 1181-1193.

[23] Chandrupatla T R, Belegundu A D, 1997, Finite Elements in Engineering, 2nd ED, *Prentice Hall.*

[24] Matthies H G, Steindorf J, 2002, Partitioned but Strongly Coupled Iteration Schemes for Nonlinear Fluid Structure Interaction, *Scientific Computing*, 2002-05.

[25] Hirt C W, Amsden A A, Cook J L, 1974, An Arbitrary Lagrangian-Eularian Computing Method for All Flow Speeds, *Journal of Computational Physics,* 14, 227.

[26] Livermore Software Technology Corporation (LSTC), 2004, www.lstc.com, *LS-DYNA 970 user manual*, Livermore, CA, USA.

[27] Le Sourne H, Couty N, Besnier F, 2003, LS-DYNA Applications in Shipbuilding, *4th European LS-DYNA User Conference*, Plenary Session II, A-11-01-16.

[28] LSTC, 2000, LS-DYNA. Training class in ALE and fluid-structure coupling. *Livermore Software Technology Corporation*. Livermore, USA.

[29] Hansbo P, Hermansson J, Svedberg T, 2003, Nitsche's method combined with space-time finite elements for ALE fluid structure interaction problems, *Chalmers Finite Element Centre*, PREPRINT 2003-14.

[30] Marzougui D, Cing-Dao Kan, Bedewi N E, 1996, Development and Validation of an NCAP simulation using LS-DYNA3D, *FHWA/NHTSA National crash Centre*, The George Washington University, Virginia, USA.

[31] Hadzic I, Mallon F, Peric M, 2001, Numerical Simulation of Sloshing, Fluent.Inc, *Application Brief EX161*, Tokyo, Japan.

[32] Snyman J A, 2000, The LFOPC leap-frog method for constrained optimization. *Computers Math. App*., 40 8/9 (2000), 1085-1096.

[33] Toolkit for Design Optimisation (TDO) v1.2a user guide, 2000. *Multidisciplinary Design Optimisation Group*, University of Pretoria, South Africa.

[34] Stander N, Egggleston T A, Craig K J, Roux W J, 2003, LS-OPT user manual v2, Design Optimisation Software for the Engineering Analyst, Livermore Software Technology Corporation, Livermore, CA.

[35] Snyman J A, 2005, Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms, *Springer*, New York, New York.

[36] Groenwold AA, Snyman JA, 2001, Global optimisation using dynamic search trajectories, *Global Optimisation*.

[37] Craig K J, Stander N, Dooge D A, Varadappa S, 2002, MDO of automotive vehicle for crashworthiness and NVH using response surface methods, AIAA paper 2002-5607, *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimizatio*n, Atlanta, GA, USA.

[38] Snyman J A, Hay A M, 2000, The Dynamic-Q Optimisation method: An alternative to SQP? *Journal of Computers and Mathematical Applications.*

[39] Craig K J, De Kock D J, Snyman J A, 1999, Using CFD and Mathematical Optimization to Investigate Air Pollution due to Stacks, *International Journal for Numerical Methods in Engineering*, Vol. 44, pp 551-565.

[40] Myers R H, Montgomery D C, 1995, Response Surface Methodology, Process and Product Optimisation using Design Experiments, *Wiley*.

[41] INGENET website, http://ingenet.ulpgc.es , NETworked INdustrial Design and Control Applications using GEnetic Algorithms and Evolution Strategies.

[42] Qu X, Haftka T, Venter G, 2003, New Formulation of a Minimum-Bias Experimental Design Based on Gauss Quadrature. Proceedings of 5th World Congress on Multidisciplinary Design Optimization, Lido di Jesolo-Venice.

[43] Stander N, Craig K J, 2002, On the robustness of a simple domain reduction scheme for simulation-based optimization. *Engineering Computations*, Vol. 19 No. 4, 431 – 450.

[44] Bishop C M, 1995, Neural Networks for Pattern Recognition, *Oxford University Press*. ISBN: 0198538642.

[45] Simpson T W, 1998, A concept exploration method for product family design. Ph. D. Thesis, *Georgia Institute of Technology*.

[46] Stander N, Roux W J, Giger M, Redhe M, Fedorova N, Haarhoff J, 2003, Crashworthiness optimization in LS-OPT: Case studies in metamodeling and random search techniques. *Proceedings of the 4th European LS-DYNA Conference*, Ulm, Germany.

[47] Jin R, Chen W, Simpson T W, 2000, Comparative studies of metamodeling techniques under multiple modelling criteria, *AIAA* Paper AIAA-2000-4801.

[48] Zang T A, Green L L, 1999, Multidisciplinary Design Optimisation Techniques: Implications and Opportunities for Fluid Dynamics Research, *AIAA*, 99-3798.

[49] Giesing J P, Barthelemy J F, 1998, A Summary of Industry MDO Applications and Needs, 7-th *AIAA/USAF/NASA/ISSMO*, St. Louis.

[50] White F M, 2003, Fluid Mechanics, 5th ed. *McGraw Hill*, USA.

[51] Craig K J, Kingsley T C, Dieterich R, Haarhoff, L J & Stander N, 2003, Design optimization of the fluid-structure interaction in a fuel tank, *16th AIAA Computational Fluid Dynamics Conference*, Orlando, FA, 23-26 June 2003.

[52] Mathworks.Inc, 2003, Matlab users guide v6, www.mathworks.com

[53] Dieterich R, 2002. Geometric Design Optimisation of a Fuel Tank in Order to Minimize Sloshing. Undergraduate thesis report, *University of Pretoria*, South Africa.

[54] Craig K J, Stander N, Dooge D A, Varadappa S, 2005, Automotive crashworthiness design using response surface-based variable screening and optimization, *Engineering Computations*, Vol.22 No.1, 38-61.

University of Pretoria etd – Kingsley, T C (2005)

APPENDICES

University of Pretoria etd – Kingsley, T C (2005)

APPENDIX A: Sample 2D GAMBIT journal file

```
/ Combined vertical and horizontal baffles
/ Created by Thomas Kingsley

solver select "FLUENT 5/6"

 $MB_cent = <<mid_baf_centroid>>
 $SB_cent = <<side_baf_centroid>>
 $MB_height = <<mid_baf_height>>
 $SB_wide = <<side_baf_width>>

/ Geometry

vertex create coordinates 0 0 0
vertex create coordinates 0 100 0
vertex create coordinates 0 200 0
edge create straight "vertex.2" "vertex.3"
edge create straight "vertex.1" "vertex.2"
vertex create coordinates ($SB_cent-$SB_wide/2) 0 0
edge create straight "vertex.1" "vertex.4"
vertex create coordinates ($SB_cent+$SB_wide/2) 0 0
edge create straight "vertex.4" "vertex.5"
face create translate "edge.2" onedge "edge.3"
face create translate "edge.6" onedge "edge.4"
face create translate "edge.1" onedge "edge.7"
face create translate "edge.12" onedge "edge.10"
vertex create coordinates 200 0 0
vertex create coordinates 200 ($MB_cent-$MB_height/2) 0
vertex create coordinates 200 ($MB_cent+$MB_height/2) 0
vertex create coordinates 200 200 0
edge create straight "vertex.16" "vertex.17"
edge create straight "vertex.15" "vertex.16"
edge create straight "vertex.14" "vertex.15"
edge create straight "vertex.5" "vertex.14"
edge create straight "vertex.13" "vertex.17"
vertex create coordinates 400 0 0
vertex create coordinates (400-($SB_cent-$SB_wide/2)) 0 0
vertex create coordinates (400-($SB_cent+$SB_wide/2)) 0 0
vertex create coordinates 400 100 0
vertex create coordinates 400 200 0
edge create straight "vertex.21" "vertex.22"
edge create straight "vertex.18" "vertex.21"
edge create straight "vertex.18" "vertex.19"
edge create straight "vertex.19" "vertex.20"
face create translate "edge.23" onedge "edge.24"
face create translate "edge.22" onedge "edge.28"
face create translate "edge.27" onedge "edge.25"
face create translate "edge.30" onedge "edge.34"
edge create straight "vertex.17" "vertex.30"
edge create straight "vertex.14" "vertex.20"
face create wireframe "edge.38" "edge.17" "edge.18" "edge.19" "edge.39" \
  "edge.33" "edge.36" real
face create wireframe "edge.20" "edge.21" "edge.15" "edge.18" "edge.9" \
  "edge.19" "edge.17" real

/ Mesh

face mesh "face.1" "face.2" "face.3" "face.4" "face.5" "face.6" "face.7" \
  "face.8" "face.9" "face.10" map size 2

/ BC

physics create "baffles" btype "WALL" edge "edge.10" "edge.18" "edge.34"

export fluent5 "2D_VF_tank.msh" nozval

abort
```

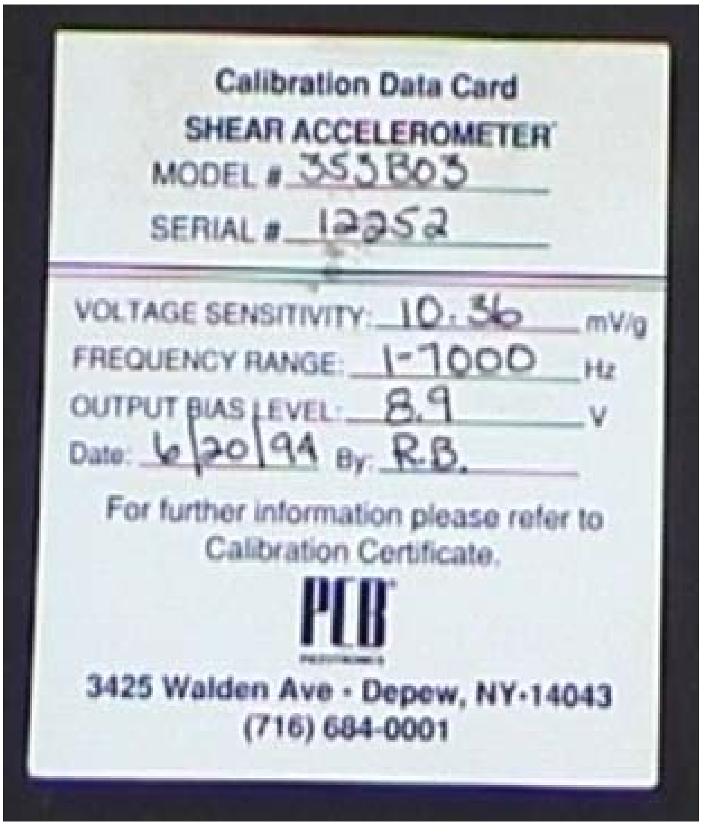
APPENDIX B: Acceleration to momentum source conversion code

```
#include "udf.h"
#include <stdio.h>
#include <stdlib.h>
#define timeend 4


real AlnrX,delt=0.015625;

DEFINE_ADJUST(Accl, domain)
{
        FILE *aptr = fopen("accelprof.txt","r");

        real t,line,lower,upper,tup,tlow;
        int i;
        char temp[20];

        t = RP_Get_Real("flow-time");

        if (t<=timeend) {
                line = (t/delt);
                for (i=0;i<=line;i++) {
                        fgets(temp, 20, aptr);
                }
                lower = atof(temp);
                fgets(temp, 20, aptr);
                upper = atof(temp);
                tup = i*delt;
                tlow = (i-1)*delt;
                AlnrX = ((tup-t)/(tup-tlow)*(upper-lower)-upper);

                /*    printf("interp = %f\n",((tup-t)/(tup-tlow)*(upper-lower)-
upper));*/
        }
        else {
                AlnrX = 0.0;
        }
        printf("time = %f\n",t);
        printf("Acceleration = %f\n",AlnrX);

        /*  printf("upper = %f\n",upper);
         *  printf("lower = %f\n",lower);
         *  printf("tup = %f\n",tup);
         *  printf("tlow = %f\n",tlow);*/
        fclose(aptr);
}


DEFINE_SOURCE(xmom, cell, thread, dS, eqn)
{
        double pos[ND_ND];
        double rho;
        double source;
        C_CENTROID(pos,cell,thread);

        rho  = C_R(cell,thread);

        dS[eqn] = 0.0;
        source = AlnrX *rho;

        return source;
}
```

APPENDIX C: Sample 2D Fluent journal file

```
f/rc "2D_VF_tank.msh"
g/s 0.001 0.001
de/m u1o y
de/m/multi vof 2 geo-reconstruct 0.25 no yes
de/m/v lam y
de/mat/cc air water y constant 998 n n y constant 0.001 n n n n n n
de/ph/pd phase-2 phase-2 y water
de/m/multi vof 2 geo-reconstruct 0.25 no yes
;de/ud/cf compile libudf y /home/thomas/2dslosh/LSOPT/LRF_udf_dp.c
de/ud/cf load ../../libudf
de/bc/fl fluid mixture y n y
"udf"
"xmom_source"
n n n y 0 0 n n
de/ud/fh
"none"
"Accl"
"none"
"none"
"none"
"none"
de/oc gr y 0 -9.81
so/se/dis-s p 13
so/se/dis-s f 22
so/se/dis-s m 0
so/se/ur p 0.8
so/se/ur d 1
so/se/ur bf 1
so/se/ur m 0.8
so/se/ur mp 1
so/in/sd/p2/mp 1
so/in if
so/se ts 0.00025
(rpsetvar 'piso/skew-iter 0)
a mir y n -1 1 <<fill_level>> 1
so/mo/re/pl y
(load "../../tui_patch.txt")
(patch (arg-patch-dom 'phase-2 'mp 0 () (list 'hexahedron-r0)))
(rpsetvar 'monitor/commands '((command-1 40 #t "su/is/phase-2 vof
free-surf () 0.5 ()") (command-2 40 #t "f/e/as data_%t free-surf () n
y yc q y") (command-3 40 #t "su/ds/free-surf")))
s/dti 8000 50
exit y
```

APPENDIX D: Accelerometer calibration certificate



Calibration Data Card

SHEAR ACCELEROMETER

MODEL # 353B03

SERIAL # 12252

VOLTAGE SENSITIVITY: 10.36 mV/g

FREQUENCY RANGE: 1-7000 Hz

OUTPUT BIAS LEVEL: 8.9 V

Date: 6/20/94 By: R.B.

For further information please refer to Calibration Certificate.

PCB

3425 Walden Ave • Depew, NY•14043
(716) 684-0001

APPENDIX E: Calibration of WIKA pressure sensor

Calibration of the WIKA pressure sensor was done using the equipment used in Figure E1 below. The equipment is the same as that which was used during the experimental testing and includes a laptop, data logger, WIKA pressure sensor, and power converter. Coloured fluid was poured into a Perspex cylinder to create the desired head of water. The level of water was then measured using a 1 meter ruler and referenced to a voltage reading that came from the equipment. Figure E2 below shows a comparison of the measured voltages compared with those expected from the pressure sensor based on the manufacturer’s provided ratings.

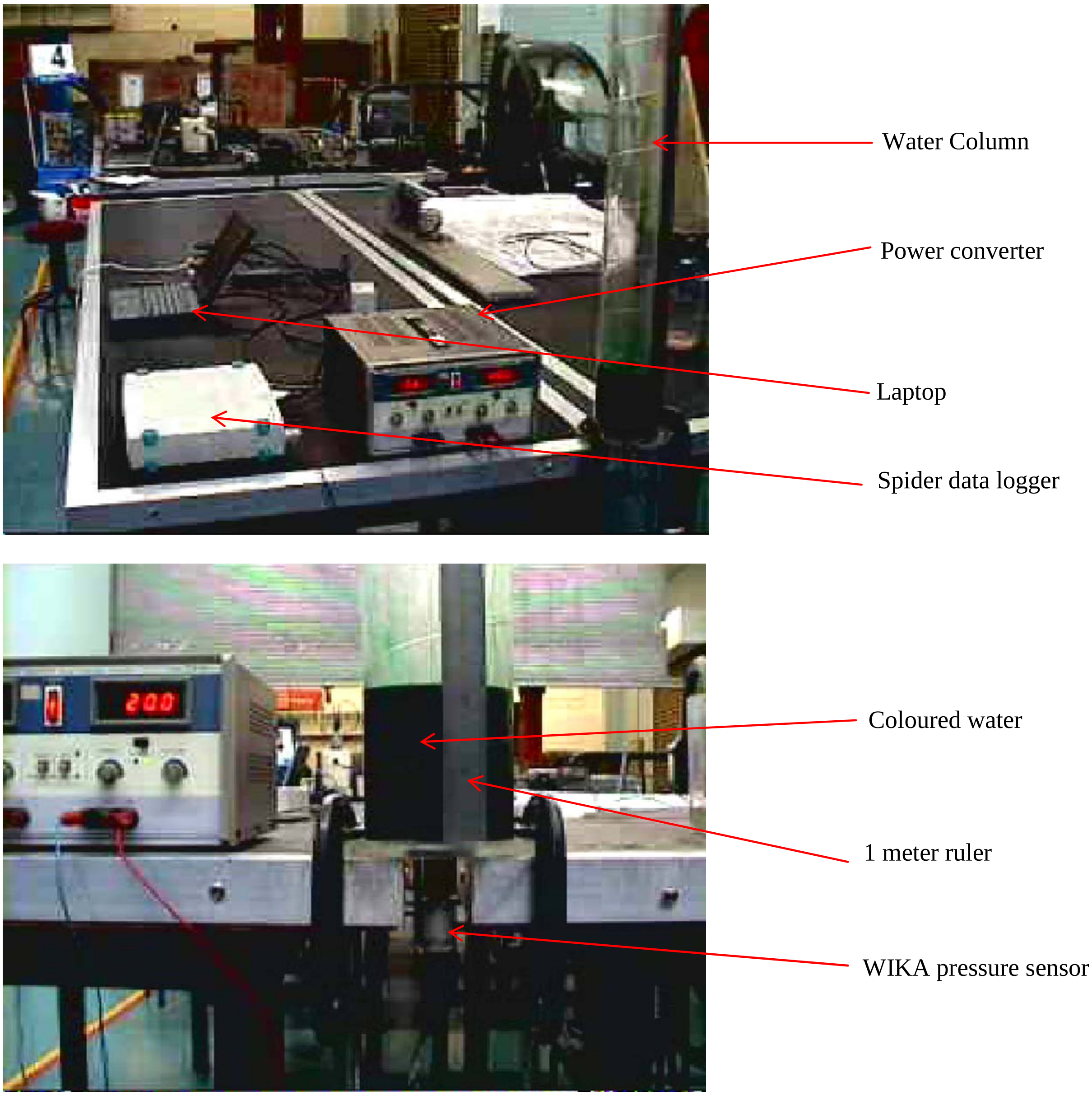


Figure E 1: WIKA pressure sensor calibration setup

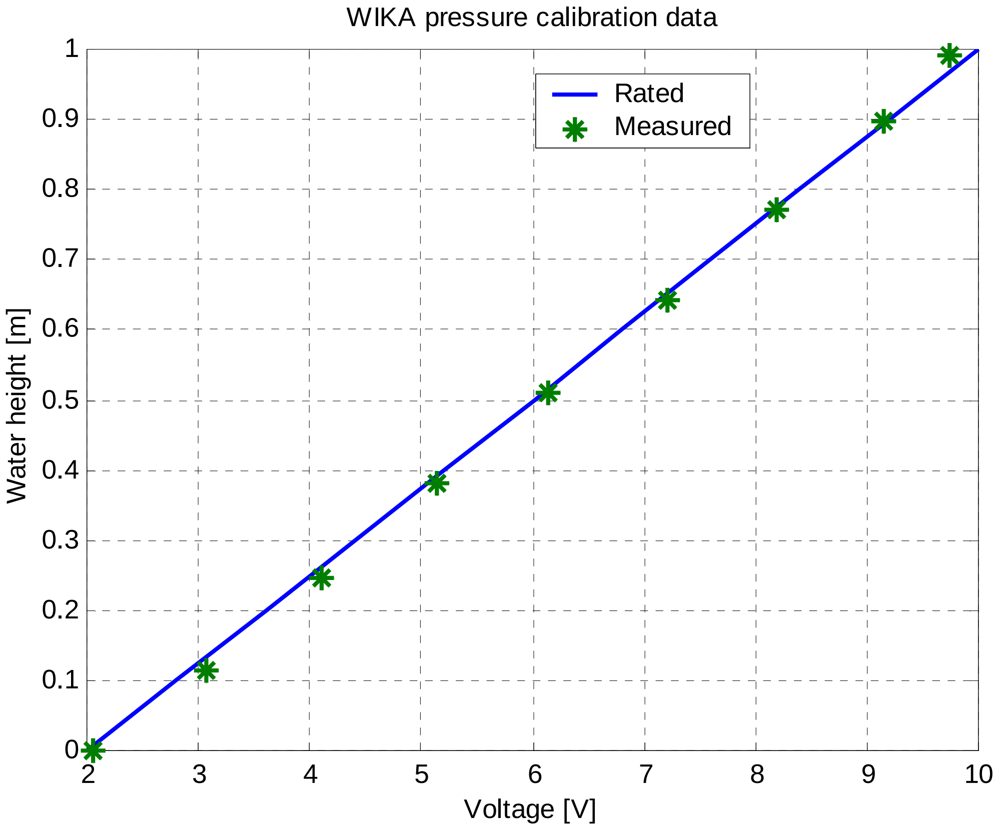


Figure E 2: Measured data versus rated voltages

Is was determined that the voltages were within acceptable limits for the purposes of the experiments. The R2 deviation from the rated line was 0.9995.

APPENDIX F: 3D optimisation LSOPT command file

```
"3D slosh opt"
Author "Rolf Deiterich"
$ Created on Fri Sep  6 00:44:41 2002
$
$ DESIGN VARIABLES
$
variables 2
 Variable 'LSholedia' 40
  Lower bound variable 'LSholedia' 15
  Upper bound variable 'LSholedia' 80
 Variable 'LSbaffleheight' 300
  Lower bound variable 'LSbaffleheight' 80
  Upper bound variable 'LSbaffleheight' 380
solvers 1
responses 6
$
$ NO HISTORIES ARE DEFINED
$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "/home/rolf/build/Rolf_fluent_script"
  solver input file "/home/rolf/build/fluentjourolf.jou"
$  prepro own
$  prepro command "gambit -inp"
$  prepro input file "/home/rolf/build/lsgridgen.jou"
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'Ay' {(LSbaffleheight-(2*LSholedia))/(1+2)}
 response 'Ay' linear
 response 'Az' {((400-(4*LSholedia))/(1+4))}
 response 'Az' linear
 response 'smallgap' {((LSbaffleheight-(2*LSholedia))/3)-(LSholedia/2)}
 response 'smallgap' linear
 response 'areamin' {(400*LSbaffleheight)-(8*3.14*LSholedia*LSholedia)}
 response 'areamin' linear
 response 'Max_Dev_from_file' 1.0 0.0 "../../fluentc.out"
 response 'Max_Dev_from_file' linear
 response 'Funnyval_from_file' 1.0 0.0 "../../funnyval.out"
 response 'Funnyval_from_file' linear
$
$ NO HISTORIES DEFINED FOR SOLVER "fluent"
$
$
$ OBJECTIVE FUNCTIONS
$
 objectives 1
 objective 'Funnyval_from_file'
$
$ CONSTRAINT DEFINITIONS
$
 constraints 4

 constraint 'areamin'
  strict
  upper bound constraint 'areamin' 95000
  slack
 constraint 'Ay'
  strict
  lower bound constraint 'Ay' 5
  slack
 constraint 'Az'
  strict
  lower bound constraint 'Az' 10
  slack
 constraint 'smallgap'
  strict
  lower bound constraint 'smallgap' 8
  slack
```

University of Pretoria etd – Kingsley, T C (2005)

```
$
$ EXPERIMENTAL DESIGN
$
 Order linear
 Experimental design dopt
 Number experiment 5
$
$ JOB INFO
$
 concurrent jobs 1
 iterate param design 0.01
 iterate param objective 0.01
 iterate 1
STOP
```

APPENDIX G: 3D baffled tank GAMBIT journal file

```
/date 2002 09 22  Time 22:28 (By Rolf Deiterich)
/substituted file

RESET
SOLVER SELECT "FLUENT 5/6"
default set "MESH.FACE.AUTO_SMOOTH" numeric 0
default set "MESH.VOLUME.AUTO_SMOOTH" numeric 0


/define tank
$tank_height=400
$tank_width=400
$tank_length=500
$tank_half_width=($tank_width/2)

/define gaps
$tank_baffle_height=187.7
$bottom_gap_height=(($tank_height-$tank_baffle_height)/2)
$top_gap_height=$bottom_gap_height

/define line coords
$line1y=0
$line2y=$bottom_gap_height
$line4y=$tank_height
$line3y=($tank_height-$top_gap_height)

/define holes
$Nholes_accross=4
$half_holes_accross=(($Nholes_accross/2))
$Nholes_up=2

$D=15
$R=($D/2)
$holez=($R*COS(45))
$holey=($R*SIN(45))
$squaremove=($R/2)

/MESH
$meshinter=3

$meshsize=3
$NHmeshsize=5


$ratio1=1.02
$ratio2=1.02
$number_of_baffles=4
$copyntimes=($number_of_baffles)
$vollength=($tank_length/($number_of_baffles+1))

/holes accross z
$Az=(($tank_width-($Nholes_accross*$D))/(1+$Nholes_accross))

/holes up y
$Ay=(($tank_baffle_height-($Nholes_up*$D))/(1+$Nholes_up))

/???????????????????????SET MESH SIZE????????????????
$Aymeshsize=5
IF COND ($Ay .LE. 48)
        $Aymeshsize=4
ENDIF

IF COND ($Ay .LE. 32)
        $Aymeshsize=3
ENDIF

IF COND ($Ay .LE. 16)
        $Aymeshsize=2
ENDIF
//
```

```
$Azmeshsize=5
IF COND ($Az .LE. 48)
        $Azmeshsize=4
ENDIF

IF COND ($Az .LE. 32)
        $Azmeshsize=3
ENDIF

IF COND ($Az .LE. 16)
        $Azmeshsize=2
ENDIF

/???????????????????????SET MESH SIZE???????????????


/create base points
vertex create "p1" coordinates 0 0 0
vertex create "p2" coordinates 0 0 $tank_half_width
vertex create "p3" coordinates 0 $bottom_gap_height 0
vertex create "p4" coordinates 0 $bottom_gap_height $tank_half_width
vertex create "p5" coordinates 0 ($tank_height-$top_gap_height) 0
vertex create "p6" coordinates 0 ($tank_height-$top_gap_height) $tank_half_width
vertex create "p7" coordinates 0 $tank_height 0
vertex create "p8" coordinates 0 $tank_height $tank_half_width


/Create the holes accross and then up
/y is up z is accross
$y=1
$z=1
DO PARA "$y" INIT 1 COND ($y .le. $Nholes_up) INCR 1
        DO PARA "$z" INIT 1 COND ($z .le. $half_holes_accross) INCR 1
                IF COND($z .EQ. 1)
                        $("zhole"+NTOS($y)+NTOS($z))=$Az+$R
                ELSE
                        $("zhole"+NTOS($y)+NTOS($z))=$("zhole"+NTOS($y)+NTOS($z-1))+$Az+$D
                ENDIF

                IF COND($y .EQ. 1)
                        $("yhole"+NTOS($y)+NTOS($z))=$bottom_gap_height+$Ay+$R
                                                                /=$bottom_gap_height+$Ay+$R
                ELSE
                        $("yhole"+NTOS($y)+NTOS($z))=$("yhole"+NTOS($y-1)+NTOS($z))+$Ay+$D
                ENDIF

                vertex create ("hv"+NTOS($z)+NTOS($y)+"cv")   coordinates 0 $("yhole"+NTOS($y)+NTOS($z))
$("zhole"+NTOS($y)+NTOS($z))
                vertex create ("hv"+NTOS($z)+NTOS($y)+"1")    coordinates 0 ($("yhole"+NTOS($y)+NTOS($z))-$R)
$("zhole"+NTOS($y)+NTOS($z))
                vertex create ("hv"+NTOS($z)+NTOS($y)+"2")    coordinates 0 ($("yhole"+NTOS($y)+NTOS($z))-
$holey)       ($("zhole"+NTOS($y)+NTOS($z))-$holez)
                vertex create ("hv"+NTOS($z)+NTOS($y)+"3")    coordinates 0 $("yhole"+NTOS($y)+NTOS($z))
($("zhole"+NTOS($y)+NTOS($z))-$R)
                vertex create ("hv"+NTOS($z)+NTOS($y)+"4")    coordinates 0
($("yhole"+NTOS($y)+NTOS($z))+$holey)       ($("zhole"+NTOS($y)+NTOS($z))-$holez)
                vertex create ("hv"+NTOS($z)+NTOS($y)+"5")    coordinates 0 ($("yhole"+NTOS($y)+NTOS($z))+$R)
$("zhole"+NTOS($y)+NTOS($z))
                vertex create ("hv"+NTOS($z)+NTOS($y)+"6")    coordinates 0
($("yhole"+NTOS($y)+NTOS($z))+$holey)       ($("zhole"+NTOS($y)+NTOS($z))+$holez)
                vertex create ("hv"+NTOS($z)+NTOS($y)+"7")    coordinates 0 $("yhole"+NTOS($y)+NTOS($z))
($("zhole"+NTOS($y)+NTOS($z))+$R)
                vertex create ("hv"+NTOS($z)+NTOS($y)+"8")    coordinates 0 ($("yhole"+NTOS($y)+NTOS($z))-
$holey)       ($("zhole"+NTOS($y)+NTOS($z))+$holez)
                vertex create ("hv"+NTOS($z)+NTOS($y)+"9")    coordinates 0 ($("yhole"+NTOS($y)+NTOS($z))-
$squaremove)   $("zhole"+NTOS($y)+NTOS($z))
                vertex create ("hv"+NTOS($z)+NTOS($y)+"10")   coordinates 0 $("yhole"+NTOS($y)+NTOS($z))
($("zhole"+NTOS($y)+NTOS($z))-$squaremove)
                vertex create ("hv"+NTOS($z)+NTOS($y)+"11")   coordinates 0
($("yhole"+NTOS($y)+NTOS($z))+$squaremove)   $("zhole"+NTOS($y)+NTOS($z))
                vertex create ("hv"+NTOS($z)+NTOS($y)+"12")   coordinates 0 $("yhole"+NTOS($y)+NTOS($z))
($("zhole"+NTOS($y)+NTOS($z))+$squaremove)
```

```
                        edge create ("he"+NTOS($z)+NTOS($y)+"1")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"1")   ("hv"+NTOS($z)+NTOS($y)+"2")
                        edge create ("he"+NTOS($z)+NTOS($y)+"2")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"2")   ("hv"+NTOS($z)+NTOS($y)+"3")
                        edge create ("he"+NTOS($z)+NTOS($y)+"3")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"3")   ("hv"+NTOS($z)+NTOS($y)+"4")
                        edge create ("he"+NTOS($z)+NTOS($y)+"4")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"4")   ("hv"+NTOS($z)+NTOS($y)+"5")
                        edge create ("he"+NTOS($z)+NTOS($y)+"5")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"5")   ("hv"+NTOS($z)+NTOS($y)+"6")
                        edge create ("he"+NTOS($z)+NTOS($y)+"6")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"6")   ("hv"+NTOS($z)+NTOS($y)+"7")
                        edge create ("he"+NTOS($z)+NTOS($y)+"7")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"7")   ("hv"+NTOS($z)+NTOS($y)+"8")
                        edge create ("he"+NTOS($z)+NTOS($y)+"8")  arc center2points  ("hv"+NTOS($z)+NTOS($y)+"cv")
("hv"+NTOS($z)+NTOS($y)+"8")   ("hv"+NTOS($z)+NTOS($y)+"1")
                        edge create ("he"+NTOS($z)+NTOS($y)+"9")  straight
("hv"+NTOS($z)+NTOS($y)+"9")   ("hv"+NTOS($z)+NTOS($y)+"10")
                        edge create ("he"+NTOS($z)+NTOS($y)+"10")  straight
("hv"+NTOS($z)+NTOS($y)+"10")   ("hv"+NTOS($z)+NTOS($y)+"11")
                        edge create ("he"+NTOS($z)+NTOS($y)+"11")  straight
("hv"+NTOS($z)+NTOS($y)+"11")   ("hv"+NTOS($z)+NTOS($y)+"12")
                        edge create ("he"+NTOS($z)+NTOS($y)+"12")  straight
("hv"+NTOS($z)+NTOS($y)+"12")   ("hv"+NTOS($z)+NTOS($y)+"9")
                        edge create ("he"+NTOS($z)+NTOS($y)+"13")  straight
("hv"+NTOS($z)+NTOS($y)+"9")   ("hv"+NTOS($z)+NTOS($y)+"1")
                        edge create ("he"+NTOS($z)+NTOS($y)+"14")  straight
("hv"+NTOS($z)+NTOS($y)+"10")   ("hv"+NTOS($z)+NTOS($y)+"3")
                        edge create ("he"+NTOS($z)+NTOS($y)+"15")  straight
("hv"+NTOS($z)+NTOS($y)+"11")   ("hv"+NTOS($z)+NTOS($y)+"5")
                        edge create ("he"+NTOS($z)+NTOS($y)+"16")  straight
("hv"+NTOS($z)+NTOS($y)+"12")   ("hv"+NTOS($z)+NTOS($y)+"7")


            ENDDO
ENDDO


/create bottom and top gap verticies

DO PARA "$z" INIT 1 COND ($z .LE. $half_holes_accross) INCR 1
            vertex create ("l1p"+NTOS($z)) coordinates 0 $line1y   $("zhole1"+NTOS($z))
            vertex create ("l2p"+NTOS($z)) coordinates 0 $line2y   $("zhole1"+NTOS($z))
            vertex create ("l3p"+NTOS($z)) coordinates 0 $line3y   $("zhole1"+NTOS($z))
            vertex create ("l4p"+NTOS($z)) coordinates 0 $line4y   $("zhole1"+NTOS($z))
ENDDO


/create vertical verticies on either side of baffle

DO PARA "$y" INIT 1 COND ($y .LE. $Nholes_up) INCR 1
            vertex create ("hlv"+NTOS($y)+"left")    coordinates 0 $("yhole"+NTOS($y)+"1")    0
            vertex create ("hlv"+NTOS($y)+"right")   coordinates 0 $("yhole"+NTOS($y)+"1")    $tank_half_width
ENDDO


/GAP

/create horisontal edges on bottom and top gap (always exist)

edge create "l1s1"                          straight "p1"                          "l1p1"
edge create ("l1s"+NTOS($half_holes_accross+1))  straight ("l1p"+NTOS($half_holes_accross)) "p2"

edge create "l2s1"                          straight "p3"                          "l2p1"
edge create ("l2s"+NTOS($half_holes_accross+1))  straight ("l2p"+NTOS($half_holes_accross)) "p4"
edge create "l3s1"                          straight "p5"                          "l3p1"
edge create ("l3s"+NTOS($half_holes_accross+1))  straight ("l3p"+NTOS($half_holes_accross)) "p6"

edge create "l4s1"                          straight "p7"                          "l4p1"
edge create ("l4s"+NTOS($half_holes_accross+1))  straight ("l4p"+NTOS($half_holes_accross)) "p8"

/need this single gap
/egde create "l1s1" straight "p1 "p2"
/egde create "l1s1" straight "p7 "p8"
```

University of Pretoria etd – Kingsley, T C (2005)

```
/create horizontal edges on bottom and top gap (if needed)

IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                edge create ("l1s"+NTOS($z+1)) straight ("l1p"+NTOS($z)) ("l1p"+NTOS($z+1))
                edge create ("l2s"+NTOS($z+1)) straight ("l2p"+NTOS($z)) ("l2p"+NTOS($z+1))
                edge create ("l3s"+NTOS($z+1)) straight ("l3p"+NTOS($z)) ("l3p"+NTOS($z+1))
                edge create ("l4s"+NTOS($z+1)) straight ("l4p"+NTOS($z)) ("l4p"+NTOS($z+1))
        ENDDO
ENDIF

/create vertical gap edges that always exist

edge create ("l5s1")                    straight "p1" "p3"
edge create ("l6s1")                    straight "p2" "p4"
edge create ("l5s"+NTOS($Nholes_up+3))  straight "p5" "p7"
edge create ("l6s"+NTOS($Nholes_up+3))  straight "p6" "p8"


/create vertical gap edges that might exist
IF COND($half_holes_accross .GE. 1)
        DO PARA "$z" INIT 1 COND ($z .LE. $half_holes_accross) INCR 1
                edge create ("bgl"+NTOS($z)) straight ("l1p"+NTOS($z)) ("l2p"+NTOS($z))
                edge create ("tgl"+NTOS($z)) straight ("l3p"+NTOS($z)) ("l4p"+NTOS($z))
        ENDDO
ENDIF

/create vertical baffle lines that always exist

edge create ("l5s2")                      straight ("p3")                          ("hlv1left")
edge create ("l5s"+NTOS($Nholes_up+2))     straight ("hlv"+NTOS($Nholes_up)+"left")    ("p5")
edge create ("l6s2")                      straight ("p4")                          ("hlv1right")
edge create ("l6s"+NTOS($Nholes_up+2))     straight ("hlv"+NTOS($Nholes_up)+"right")   ("p6")

/create vertical baffle line
IF COND($half_holes_accross .GT. 1)
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                edge create ("l5s"+NTOS($y+2))   straight ("hlv"+NTOS($y)+"left")  ("hlv"+NTOS($y+1)+"left")
                edge create ("l6s"+NTOS($y+2))   straight ("hlv"+NTOS($y)+"right") ("hlv"+NTOS($y+1)+"right")
        ENDDO
ENDIF
/create internal baffle lines v bottom

DO PARA "$z" INIT 1 COND ($z .LE. $half_holes_accross) INCR 1
        edge create ("lbl"+NTOS($z)) straight ("l2p"+NTOS($z)) ("hv"+NTOS($z)+"11")
ENDDO

/create internal baffle lines v midle
IF COND($half_holes_accross .GT. 1)
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up)
                DO PARA "$z" INIT 1 COND ($z .LE. $half_holes_accross) INCR 1
                        edge create ("bivl"+NTOS($y)+NTOS($z)) straight ("hv"+NTOS($z)+NTOS($y)+"5")
("hv"+NTOS($z)+NTOS($y+1)+"1")
                ENDDO
        ENDDO
ENDIF

/create internal baffle lines v top
DO PARA "$z" INIT 1 COND ($z .LE. $half_holes_accross) INCR 1
        edge create ("ubl"+NTOS($z)) straight ("hv"+NTOS($z)+NTOS($Nholes_up)+"5") ("l3p"+NTOS($z))
ENDDO

/create internal baffle lines h left
DO PARA "$y" INIT 1 COND ($y .LE. $Nholes_up) INCR 1
        edge create ("lhbl"+NTOS($y)) straight ("hlv"+NTOS($y)+"left") ("hv1"+NTOS($y)+"3")
ENDDO


/create internal baffle lines h middle
IF COND($half_holes_accross .GT. 1)
        DO PARA "$y" INIT 1 COND ($y .LE. $Nholes_up)
                DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                        edge create ("bihl"+NTOS($y)+NTOS($z)) straight ("hv"+NTOS($z)+NTOS($y)+"7")
("hv"+NTOS($z+1)+NTOS($y)+"3")
                ENDDO
        ENDDO
ENDIF

/create internal baffle lines h right
DO PARA "$y" INIT 1 COND ($y .LE. $Nholes_up) INCR 1
        edge create ("rhbl"+NTOS($y)) straight ("hv"+NTOS($half_holes_accross)+NTOS($y)+"7")
("hlv"+NTOS($y)+"right")
ENDDO

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
/FACES
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/create faces on bottom gap
/lower gap face

face create "lgfleft"   wireframe "l1s1" "l5s1" "l2s1" "bgl1"
face create "lgfright"  wireframe ("l1s"+NTOS($half_holes_accross+1)) ("bgl"+NTOS($half_holes_accross))
("l2s"+NTOS($half_holes_accross+1)) "l6s1"

IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face create ("lgf"+NTOS($z)) wireframe ("l1s"+NTOS($z+1)) ("bgl"+NTOS($z)) ("l2s"+NTOS($z+1))
("bgl"+NTOS($z+1))
        ENDDO
ENDIF

/create faces on top gap
/upper gap face

face create "ugfleft"   wireframe ("l3s1") ("l5s"+NTOS($Nholes_up+3)) "l4s1" "tgl1"
face create "ugfright"  wireframe ("l3s"+NTOS($half_holes_accross+1)) ("tgl"+NTOS($half_holes_accross))
("l4s"+NTOS($half_holes_accross+1)) ("l6s"+NTOS($Nholes_up+3))

IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face create ("ugf"+NTOS($z)) wireframe ("l3s"+NTOS($z+1)) ("tgl"+NTOS($z)) ("l4s"+NTOS($z+1))
("tgl"+NTOS($z+1))
        ENDDO
ENDIF

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/create lower baffle faces that always exist

face create "bf11" wireframe "l2s1" "l5s2" "lhbl1" "he112" "he111" "lbl1"
face create ("bf1"+NTOS($half_holes_accross+1)) wireframe ("l2s"+NTOS($half_holes_accross+1))
("lbl"+NTOS($half_holes_accross)) ("he"+NTOS($half_holes_accross)+"18") ("he"+NTOS($half_holes_accross)+"17") "rhbl1"
"l6s2"

/create lower baffle faces that might exist

IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face create ("bf1"+NTOS($z+1)) wireframe ("l2s"+NTOS($z+1)) ("lbl"+NTOS($z))
("he"+NTOS($z)+"18") ("he"+NTOS($z)+"17") ("bihl1"+NTOS($z)) ("he"+NTOS($z+1)+"12") ("he"+NTOS($z+1)+"11")
("lbl"+NTOS($z+1))
        ENDDO
ENDIF

/create upper baffle faces that always exist

face create ("bf"+NTOS($Nholes_up+1)+"1") wireframe ("l3s1") ("l5s"+NTOS($Nholes_up+2)) ("lhbl"+NTOS($Nholes_up))
("he1"+NTOS($Nholes_up)+"3") ("he1"+NTOS($Nholes_up)+"4") "ubl1"
face create ("bf"+NTOS($Nholes_up+1)+NTOS($half_holes_accross+1)) wireframe ("l3s"+NTOS($half_holes_accross+1))
("ubl"+NTOS($half_holes_accross)) ("he"+NTOS($half_holes_accross)+NTOS($Nholes_up)+"5")
("he"+NTOS($half_holes_accross)+NTOS($Nholes_up)+"6") ("rhbl"+NTOS($Nholes_up)) ("l6s"+NTOS($Nholes_up+2))

/create upper baffle faces that might exist

IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face create ("bf"+NTOS($Nholes_up+1)+NTOS($z+1)) wireframe ("l3s"+NTOS($z+1))
("ubl"+NTOS($z)) ("he"+NTOS($z)+NTOS($Nholes_up)+"5") ("he"+NTOS($z)+NTOS($Nholes_up)+"6")
```

```
("bihl"+NTOS($Nholes_up)+NTOS($z)) ("he"+NTOS($z+1)+NTOS($Nholes_up)+"3")
("he"+NTOS($z+1)+NTOS($Nholes_up)+"4") ("ubl"+NTOS($z+1))
        ENDDO
ENDIF


/LONG IF

IF COND($half_holes_accross .GT. 1)

        /create middle faces left
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                face create ("bf"+NTOS($y+1)+"1") wireframe ("lhbl"+NTOS($y)) ("l5s"+NTOS($y+2))
("lhbl"+NTOS($y+1)) ("he1"+NTOS($y+1)+"2") ("he1"+NTOS($y+1)+"1") ("bivl"+NTOS($y)+"1") ("he1"+NTOS($y)+"4")
("he1"+NTOS($y)+"3")
        ENDDO

        /create middle faces right
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                face create ("bf"+NTOS($y+1)+NTOS($half_holes_accross+1)) wireframe ("rhbl"+NTOS($y))
("l6s"+NTOS($y+2)) ("rhbl"+NTOS($y+1)) ("he"+NTOS($half_holes_accross)+NTOS($y+1)+"8")
("he"+NTOS($half_holes_accross)+NTOS($y+1)+"7") ("bivl"+NTOS($y)+NTOS($half_holes_accross))
("he"+NTOS($half_holes_accross)+NTOS($y)+"5") ("he"+NTOS($half_holes_accross)+NTOS($y)+"6")
        ENDDO


        /create middle faces rest
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross)
                        face create ("bf"+NTOS($y+1)+NTOS($z+1)) wireframe ("bihl"+NTOS($y)+NTOS($z))
("he"+NTOS($z)+NTOS($y)+"6") ("he"+NTOS($z)+NTOS($y)+"5") ("bivl"+NTOS($y)+NTOS($z))
("he"+NTOS($z)+NTOS($y+1)+"8") ("he"+NTOS($z)+NTOS($y+1)+"7") ("bihl"+NTOS($y+1)+NTOS($z))
("he"+NTOS($z+1)+NTOS($y+1)+"2") ("he"+NTOS($z+1)+NTOS($y+1)+"1") ("bivl"+NTOS($y)+NTOS($z+1))
("he"+NTOS($z+1)+NTOS($y)+"4") ("he"+NTOS($z+1)+NTOS($y)+"3")

                ENDDO
        ENDDO
ENDIF


//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
/create hole faces
DO PARA "$y" INIT 1 COND ($y .LE. $Nholes_up) INCR 1
        DO PARA "$z" INIT 1 COND ($z .LE. $half_holes_accross ) INCR 1
                face create ("hf"+NTOS($y)+NTOS($z)+"1") wireframe ("he"+NTOS($z)+NTOS($y)+"1")
("he"+NTOS($z)+NTOS($y)+"2")  ("he"+NTOS($z)+NTOS($y)+"14") ("he"+NTOS($z)+NTOS($y)+"9")
("he"+NTOS($z)+NTOS($y)+"13")
                face create ("hf"+NTOS($y)+NTOS($z)+"2") wireframe ("he"+NTOS($z)+NTOS($y)+"14")
("he"+NTOS($z)+NTOS($y)+"3")  ("he"+NTOS($z)+NTOS($y)+"4")  ("he"+NTOS($z)+NTOS($y)+"15")
("he"+NTOS($z)+NTOS($y)+"10")
                face create ("hf"+NTOS($y)+NTOS($z)+"3") wireframe ("he"+NTOS($z)+NTOS($y)+"16")
("he"+NTOS($z)+NTOS($y)+"11") ("he"+NTOS($z)+NTOS($y)+"15") ("he"+NTOS($z)+NTOS($y)+"5")
("he"+NTOS($z)+NTOS($y)+"6")
                face create ("hf"+NTOS($y)+NTOS($z)+"4") wireframe ("he"+NTOS($z)+NTOS($y)+"7")
("he"+NTOS($z)+NTOS($y)+"8")  ("he"+NTOS($z)+NTOS($y)+"13") ("he"+NTOS($z)+NTOS($y)+"12")
("he"+NTOS($z)+NTOS($y)+"16")
                face create ("hf"+NTOS($y)+NTOS($z)+"5") wireframe ("he"+NTOS($z)+NTOS($y)+"9")
("he"+NTOS($z)+NTOS($y)+"10") ("he"+NTOS($z)+NTOS($y)+"11") ("he"+NTOS($z)+NTOS($y)+"12")

        ENDDO
ENDDO

/????????????????????????????????????????????????????????????????????????????????
/START OF MESHING
/????????????????????????????????????????????????????????????????????????????????

vertex create "volv1"   coordinates 0          0 -20
vertex create "volv2"   coordinates $vollength  0 -20
edge create   "voledge" straight    "volv1" "volv2"


/????????????????????????????????????????????????????????????????????????????????
/MODIFY FACE CORNERS
/????????????????????????????????????????????????????????????????????????????????
```

```
/lower baffle faces corner modify

face modify "bf11" corner "hv112"
face modify ("bf1"+NTOS($half_holes_accross+1)) corner ("hv"+NTOS($half_holes_accross)+"18")

IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face modify ("bf1"+NTOS($z+1)) corner ("hv"+NTOS($z)+"18") ("hv"+NTOS($z+1)+"12")
        ENDDO
ENDIF

/upper baffle faces corner modify

face modify ("bf"+NTOS($Nholes_up+1)+"1")                corner ("hv1"+NTOS($Nholes_up)+"4")
face modify ("bf"+NTOS($Nholes_up+1)+NTOS($half_holes_accross+1)) corner
("hv"+NTOS($half_holes_accross)+NTOS($Nholes_up)+"6")

IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face modify ("bf"+NTOS($Nholes_up+1)+NTOS($z+1)) corner
("hv"+NTOS($z)+NTOS($Nholes_up)+"6")  ("hv"+NTOS($z+1)+NTOS($Nholes_up)+"4")
        ENDDO
ENDIF


/Middle baffle faces corner modify    all if

IF COND($half_holes_accross .GT. 1)


        /middle faces left
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                face modify ("bf"+NTOS($y+1)+"1") corner ("hv1"+NTOS($y)+"4") ("hv1"+NTOS($y+1)+"2")
        ENDDO

        /middle faces right
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                face modify ("bf"+NTOS($y+1)+NTOS($half_holes_accross+1)) corner
("hv"+NTOS($half_holes_accross)+NTOS($y)+"6") ("hv"+NTOS($half_holes_accross)+NTOS($y+1)+"8")
        ENDDO

        /middle faces middle
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                        face modify ("bf"+NTOS($y+1)+NTOS($z+1)) corner ("hv"+NTOS($z)+NTOS($y)+"6")
("hv"+NTOS($z+1)+NTOS($y)+"4") ("hv"+NTOS($z)+NTOS($y+1)+"8")("hv"+NTOS($z+1)+NTOS($y+1)+"2")
                ENDDO
        ENDDO
ENDIF
```

University of Pretoria etd – Kingsley, T C (2005)

```
/????????????????????????????????????????????????????????????????????????????????
/MESH HOLES    :    Seed edges ; Mesh faces ; Create volumes : Copy volumes
/????????????????????????????????????????????????????????????????????????????????

DO PARA "$y" INIT 1 COND ($y .LE. $Nholes_up) INCR 1
        DO PARA "$z" INIT 1 COND ($z .LE. $half_holes_accross) INCR 1

                edge mesh  ("he"+NTOS($z)+NTOS($y)+"1") ("he"+NTOS($z)+NTOS($y)+"2")
("he"+NTOS($z)+NTOS($y)+"3") ("he"+NTOS($z)+NTOS($y)+"4") ("he"+NTOS($z)+NTOS($y)+"5")
("he"+NTOS($z)+NTOS($y)+"6") ("he"+NTOS($z)+NTOS($y)+"7") ("he"+NTOS($z)+NTOS($y)+"8")
("he"+NTOS($z)+NTOS($y)+"13") size $meshsize
                face mesh  ("hf"+NTOS($y)+NTOS($z)+"1") ("hf"+NTOS($y)+NTOS($z)+"2")
("hf"+NTOS($y)+NTOS($z)+"3") ("hf"+NTOS($y)+NTOS($z)+"4") submap size $meshsize
                face mesh  ("hf"+NTOS($y)+NTOS($z)+"5") map


/face mesh ("hf"+NTOS($y)+NTOS($z)+"5") map size $meshsize

                /create volumes
                volume create ("hvol"+NTOS($y)+NTOS($z)+"1") translate ("hf"+NTOS($y)+NTOS($z)+"1") onedge
"voledge"
                volume create ("hvol"+NTOS($y)+NTOS($z)+"2") translate ("hf"+NTOS($y)+NTOS($z)+"2") onedge
"voledge"
                volume create ("hvol"+NTOS($y)+NTOS($z)+"3") translate ("hf"+NTOS($y)+NTOS($z)+"3") onedge
"voledge"
                volume create ("hvol"+NTOS($y)+NTOS($z)+"4") translate ("hf"+NTOS($y)+NTOS($z)+"4") onedge
"voledge"
                volume create ("hvol"+NTOS($y)+NTOS($z)+"5") translate ("hf"+NTOS($y)+NTOS($z)+"5") onedge
"voledge"

                face cmove ("hf"+NTOS($y)+NTOS($z)+"1") ("hf"+NTOS($y)+NTOS($z)+"2")
("hf"+NTOS($y)+NTOS($z)+"3") ("hf"+NTOS($y)+NTOS($z)+"4") multiple 1 unlinkmesh offset $vollength 0 0
                face cmove ("hf"+NTOS($y)+NTOS($z)+"5") multiple 1 unlinkmesh offset $vollength 0 0

                face connect real

                volume mesh  ("hvol"+NTOS($y)+NTOS($z)+"1") cooper size $NHmeshsize
                volume mesh  ("hvol"+NTOS($y)+NTOS($z)+"2") cooper size $NHmeshsize
                volume mesh  ("hvol"+NTOS($y)+NTOS($z)+"3") cooper size $NHmeshsize
                volume mesh  ("hvol"+NTOS($y)+NTOS($z)+"4") cooper size $NHmeshsize
                volume mesh  ("hvol"+NTOS($y)+NTOS($z)+"5") cooper size $NHmeshsize

                volume cmove ("hvol"+NTOS($y)+NTOS($z)+"1") multiple $copyntimes unlinkmesh offset $vollength 0
0
                volume cmove ("hvol"+NTOS($y)+NTOS($z)+"2") multiple $copyntimes unlinkmesh offset $vollength 0
0
                volume cmove ("hvol"+NTOS($y)+NTOS($z)+"3") multiple $copyntimes unlinkmesh offset $vollength 0
0
                volume cmove ("hvol"+NTOS($y)+NTOS($z)+"4") multiple $copyntimes unlinkmesh offset $vollength 0
0
                volume cmove ("hvol"+NTOS($y)+NTOS($z)+"5") multiple $copyntimes unlinkmesh offset $vollength 0
0



        ENDDO
ENDDO

/SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
/Mesh seed the straight baffle lines
/SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

/lower baffle left
edge modify ("lhbl1") backward
edge mesh ("lhbl1") successive ratio1 $ratio1 size $Azmeshsize

edge modify ("lbl1") backward
edge mesh ("lbl1")  successive ratio1 $ratio1 size $Aymeshsize

/lower baffle right

edge mesh ("rhbl1") successive ratio1 $ratio1 size $Azmeshsize
/edge mesh ("rhbl1") size $meshsize

IF COND($half_holes_accross .GT. 1)
```

```
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                /edge mesh ("bihl1"+NTOS($z)) size $meshsize
                /edge mesh ("lbl"+NTOS($z+1)) size $meshsize
                edge mesh ("bihl1"+NTOS($z)) successive ratio1 $ratio1 ratio2 $ratio2 size $Azmeshsize
                edge modify ("lbl"+NTOS($z+1)) backward
                edge mesh ("lbl"+NTOS($z+1)) successive ratio1 $ratio1 size $Aymeshsize
        ENDDO
ENDIF


/upper baffle left
/edge mesh ("lhbl"+NTOS($Nholes_up)) size $meshsize   /do not need as the lower mesh will take care of it

//edge mesh ("ubl1") size $meshsize
edge mesh ("ubl1")  successive ratio1 $ratio1 size $Aymeshsize


/upper baffle right
/edge mesh ("rhbl"+NTOS($Nholes_up)) size $meshsize

/upper baffle middle
IF COND($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                /edge mesh ("bihl"+NTOS($Nholes_up)+NTOS($z))  size $meshsize
                /edge mesh ("ubl"+NTOS($z+1))                 size $meshsize
                edge mesh ("bihl"+NTOS($Nholes_up)+NTOS($z)) successive ratio1 $ratio1 ratio2 $ratio2 size
$Azmeshsize
                /edge modify ("ubl"+NTOS($z+1)) forward
                edge mesh ("ubl"+NTOS($z+1)) successive ratio1 $ratio1 size $Aymeshsize
        ENDDO
ENDIF


/Middle faces all if

IF COND($half_holes_accross .GT. 1)

        /middle faces left
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                edge modify ("lhbl"+NTOS($y+1)   ) backward
                edge mesh ("lhbl"+NTOS($y+1)   )    successive ratio1 $ratio1 size $Azmeshsize
                edge mesh ("bivl"+NTOS($y)+"1" )    successive ratio1 $ratio1 ratio2 $ratio2 size $Aymeshsize
        ENDDO

        /middle faces right
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                edge mesh ("rhbl"+NTOS($y+1))       successive ratio1 $ratio1 size $Azmeshsize
        ENDDO

        /middle faces middle
        DO PARA "$y" INIT 1 COND ($y .LT. $Nholes_up) INCR 1
                DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                        edge mesh ("bihl"+NTOS($y+1)+NTOS($z)) successive ratio1 $ratio1 ratio2 $ratio2 size
$Azmeshsize
                        edge mesh ("bivl"+NTOS($y)+NTOS($z+1)) successive ratio1 $ratio1 ratio2 $ratio2 size
$Aymeshsize
                ENDDO
        ENDDO
ENDIF


/MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
M
/MESH BAFFLE SQUARES
/MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
M

/face mesh     ("bf"+NTOS($y)+NTOS($z)) submap size $NHmeshsize
DO PARA "$y" INIT 1 COND ($y .LE. ($Nholes_up+1)) INCR 1
        DO PARA "$z" INIT 1 COND ($z .LE. ($half_holes_accross+1)) INCR 1
                face mesh     ("bf"+NTOS($y)+NTOS($z)) submap
                volume create ("bv"+NTOS($y)+NTOS($z)) translate ("bf"+NTOS($y)+NTOS($z)) onedge "voledge"
                face cmove    ("bf"+NTOS($y)+NTOS($z)) multiple 1 unlinkmesh offset $vollength 0 0
                face connect real
                volume mesh   ("bv"+NTOS($y)+NTOS($z)) cooper size $NHmeshsize
        volume cmove  ("bv"+NTOS($y)+NTOS($z)) multiple $copyntimes unlinkmesh offset $vollength 0 0


        ENDDO
ENDDO


/MMMMMMMMMMMMMMMMMMMMMMMMMMMMGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
/MESH THE GAPS
/MMMMMMMMMMMMMMMMMMMMMMMMMMMMGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG

/MESH THE LOWER GAP LEFT

        edge mesh      "l5s1" size $NHmeshsize
        face mesh      "lgfleft" map size $NHmeshsize
        volume create  "lgvl" translate "lgfleft" onedge "voledge"
        face cmove     "lgfleft" multiple 1 unlinkmesh offset $vollength 0 0
        face connect real
        volume mesh    "lgvl" map size $NHmeshsize
        volume cmove   "lgvl" multiple $copyntimes unlinkmesh offset $vollength 0 0


/MESH MIDDLE VOLS

IF COND ($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face mesh     ("lgf"+NTOS($z)) map size $NHmeshsize
                volume create ("lgv"+NTOS($z)) translate ("lgf"+NTOS($z)) onedge "voledge"
                face cmove    ("lgf"+NTOS($z)) multiple 1 unlinkmesh offset $vollength 0 0
                face connect real
                volume mesh   ("lgv"+NTOS($z)) map size $NHmeshsize
                volume cmove  ("lgv"+NTOS($z)) multiple $copyntimes unlinkmesh offset $vollength 0 0


        ENDDO
ENDIF

/MESH THE LOWER GAP LEFT
        face mesh      "lgfright" map size $NHmeshsize
        volume create  "lgvr" translate "lgfright" onedge "voledge"
        face cmove     "lgfright" multiple 1 unlinkmesh offset $vollength 0 0
        face connect real
        volume mesh    "lgvr" map size $NHmeshsize
        volume cmove   "lgvr" multiple $copyntimes unlinkmesh offset $vollength 0 0



/MMMMMMMMMMMMMMMMMMM GGGGGGGGGGGGGGGGGG
/MESH UPPER GAP
/MMMMMMMMMMMMMMMMMMM GGGGGGGGGGGGGGGGGG

/MESH THE UPPER GAP LEFT
        edge mesh      ("l5s"+NTOS($Nholes_up+3)) size $NHmeshsize
        face mesh      "ugfleft" map size $NHmeshsize
        volume create  "ugvl" translate "ugfleft" onedge "voledge"
        face cmove     "ugfleft" multiple 1 unlinkmesh offset $vollength 0 0
        face connect real
        volume mesh    "ugvl" map size $NHmeshsize
        volume cmove   "ugvl" multiple $copyntimes unlinkmesh offset $vollength 0 0


/MESH MIDDLE VOLS

IF COND ($half_holes_accross .GT. 1)
        DO PARA "$z" INIT 1 COND ($z .LT. $half_holes_accross) INCR 1
                face mesh     ("ugf"+NTOS($z)) map size $NHmeshsize
                volume create ("ugv"+NTOS($z)) translate ("ugf"+NTOS($z)) onedge "voledge"
                face cmove    ("ugf"+NTOS($z)) multiple 1 unlinkmesh offset $vollength 0 0
                face connect real
                volume mesh   ("ugv"+NTOS($z)) map size $NHmeshsize
                volume cmove  ("ugv"+NTOS($z)) multiple $copyntimes unlinkmesh offset $vollength 0 0

        ENDDO
ENDIF
```

```
/MESH THE UPPER GAP RIGHT
		face mesh     "ugfright" map size $NHmeshsize
		volume create "ugvr" translate "ugfright" onedge "voledge"
		face cmove    "ugfright" multiple 1 unlinkmesh offset $vollength 0 0
		face connect real
		volume mesh   "ugvr" map size $NHmeshsize
		volume cmove  "ugvr" multiple $copyntimes unlinkmesh offset $vollength 0 0

/vertex connect real
/edge connect real
face connect real


$lastvolid=lastid(t_vo)
$counterv=1

DO PARA "$counterv" INIT 1 COND ($counterv .LE. $lastvolid) INCR 1
		volume mesh ("volume."+NTOS($counterv)) submap size $NHmeshsize
ENDDO




$lastvolid=lastid(t_vo)
$counterv=1

DO PARA "$counterv" INIT 1 COND ($counterv .LE. $lastvolid) INCR 1
		volume mesh ("volume."+NTOS($counterv)) submap size $meshsize
ENDDO

/BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
/Set Boundary Conditions
/BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

/loop for setting the boundary conditions

declare $facecoords [1:3]

$lastfaceid=lastid(t_fa)
$counter=1
$xmin=10
$xmax=($tank_length-10)
/$ymin=($bottom_gap_height)
/$ymax=($tank_height-$top_gap_height)
$ymin=0
$ymax=0

/check to see if you need to create groups before you can add info to them
vertex create "phantom1" coordinates 0 0 -20
vertex create "phantom2" coordinates 0 5 -20
vertex create "phantom3" coordinates 0 5 -25
vertex create "phantom4" coordinates 0 0 -25
edge create "phe1" straight "phantom1" "phantom2"
edge create "phe2" straight "phantom2" "phantom3"
edge create "phe3" straight "phantom3" "phantom4"
edge create "phe4" straight "phantom4" "phantom1"
face create "phantomface1" wireframe "phe1" "phe2" "phe3" "phe4"

vertex create "phantom5" coordinates 0 0 -40
vertex create "phantom6" coordinates 0 5 -40
vertex create "phantom7" coordinates 0 5 -45
vertex create "phantom8" coordinates 0 0 -45
edge create "phe5" straight "phantom5" "phantom6"
edge create "phe6" straight "phantom6" "phantom7"
edge create "phe7" straight "phantom7" "phantom8"
edge create "phe8" straight "phantom8" "phantom5"
face create "phantomface2" wireframe "phe5" "phe6" "phe7" "phe8"

group create "wallsym"  face "phantomface1"
group create "intwalls" face "phantomface2"

$Shift=2
$z=0
$y=0
```

```
DO PARA "$counter" INIT 1 COND ($counter .LE. $lastfaceid) INCR 1
    $facecoords=ENT2LOC("face."+NTOS($counter))

    IF COND ($facecoords[3] .EQ. $tank_half_width)
        group add "wallsym" face ("face."+NTOS($counter))
    ENDIF

    DO PARA "$z" INIT 1 COND ($z .LE. ($half_holes_accross+1)) INCR 1

        DO PARA "$y" INIT 1 COND ($y .LE. ($Nholes_up+1)) INCR 1

            IF COND ($z .EQ. 1)
                $zmin=0
                $zmax=($("zhole1"+NTOS($z))-$Shift)
            ENDIF

            IF COND (($z .GT. 1) .AND. ($z .LE. ($half_holes_accross)))
                $zmin=($("zhole1"+NTOS($z-1))+$Shift)
                $zmax=($("zhole1"+NTOS($z))-$Shift)
            ENDIF

            IF COND ( $z .EQ. ($half_holes_accross+1) )
                $zmin=($("zhole1"+NTOS($z-1))+$Shift)
                $zmax=$tank_half_width
            ENDIF



            IF COND ($y .EQ. 1)
                $ymin=($bottom_gap_height+5)
                $ymax=($("yhole"+NTOS($y)+"1")-$R)
            ENDIF

            IF COND (($y .GT. 1) .AND. ($y .LT. ($Nholes_up+1)))
                $ymin=($("yhole"+NTOS($y-1)+"1")+$R)
                $ymax=($("yhole"+NTOS($y)+"1")-$R)
            ENDIF

            IF COND ($y .EQ. ($Nholes_up+1))
                $ymin=($("yhole"+NTOS($y-1)+"1")+$R)
                $ymax=($tank_height-$top_gap_height-5)

            ENDIF



            IF COND ( ($facecoords[1] .GT. $xmin) .AND. ($facecoords[1] .LT. $xmax) .AND.
($facecoords[2] .GT. $ymin) .AND. ($facecoords[2] .LT. $ymax) .AND. ($facecoords[3] .GT. $zmin) .AND.
($facecoords[3] .LT. $zmax))
                group add "intwalls" face ("face."+NTOS($counter))

            ENDIF
        ENDDO
    ENDDO

ENDDO

edge delete "voledge" lowertopology
face delete "phantomface1" lowertopology
face delete "phantomface2" lowertopology

/check to see if this is valid
physics create "symwallplane"  btype "SYMMETRY" group "wallsym"
physics create "internalwalls" btype "WALL"     group "intwalls"

default set "FILE_IO.FLUENT5.EXPORT_USING_UTILITY" numeric 1
/export fluent5 "/home/rolf/gm1.msh"
export fluent5 "lsmesh.msh"


//EOF
```

University of Pretoria etd – Kingsley, T C (2005)

APPENDIX H: 3D baffled tank Fluent journal file

```
!3D sloshing Fluent setup journal
f/rc lsmesh.msh
g/s 0.001 0.001 0.001
!echo grid/check
!echo display/grid-outline
!echo display/view dv
d/v read-v
"viewr.vw"
de/m u1o y
de/m/solver seg y
de/m multi vof 2 geo-reconstruct 0.25 no yes
de/m/v lam y
de/mat/cc air water y constant 998 n n y constant 0.001 n n n n n n
de/ph pd phase-2 y water
de/m multi vof 2 geo-reconstruct 0.25 no yes
de/ud cf libudf
de/bc/fl
fluid
mixture
y
y
"udf"
"xmom_source"
n
0
n
0
n
y
0
0
0
0
0
1
n
de/ud/fh
"none"
"Accl"
"none"
"none"
"none"
"none"
!echo de/mat/cc air water y constant 998 n n y constant 0.001 n n n n n n
!echo de/ph pd phase-2 y water
de/oc grav y 0 -9.81 0
so/se/dis-s p 13
so/se/dis-s f 22
so/se/dis-s m 0
so/se/ur p 0.6
so/se/ur d 1
so/se/ur bf 1
so/se/ur m 0.8
so/se/ur mp 1
so/in/cd az
so/in if
so/se ts 0.001
(rpsetvar 'piso/skew? #f)
a mih y n 0 0.5 0 0.2 0 0.2
(cx-gui-do cx-activate-item "MenuBar*InitializeSubMenu*Patch...")
(cx-gui-do cx-set-list-selections "Patch*Frame1*Frame2*List2(Variable)" '( 4))
(cx-gui-do cx-activate-item "Patch*Frame1*Frame2*List2(Variable)")
(cx-gui-do cx-set-real-entry-list "Patch*Frame2*RealEntry1(Value)" '( 1))
(cx-gui-do cx-set-list-selections "Patch*Frame3*Frame2*List2(Registers To Patch)" '( 0))
(cx-gui-do cx-activate-item "Patch*Frame3*Frame2*List2(Registers To Patch)")
(cx-gui-do cx-activate-item "Patch*PanelButtons*PushButton1(Patch)")
(cx-gui-do cx-activate-item "Patch*PanelButtons*PushButton1(Close)")
so/mo/re/pl y
!echo de/mat/ copy fluid water-liquid
!echo de/ph pd phase-2 y water-liquid
```

University of Pretoria etd – Kingsley, T C (2005)

```
!echo (rpsetvar 'monitor/commands '((command-1 10 #t "su/is/vof-phase-2 free-surf () 0.5 ()") (command-2 10 #t "f/e/as data_%t free-surf () n y yc q y") (command-3 10 #t "su/ds/free-surf")))
(rpsetvar 'monitor/commands '((command-1 20 #t "su/is/vof-phase-2 free-surf () 0.5 ()") (command-2 20 #t "f/e/as data_%t free-surf () n y yc q y") (command-3 20 #t "su/ds/free-surf")))
!echo so/ dti 4 20
f/wc lsmesh.cas.gz

f/as/ rn lsmesh.gz
f/as/ df 500
f/as/ cf 500
so dti 7000 20
```

APPENDIX I: 3D data extraction source code

```
// Create by Rolf Deitrich
// 3D Data extraction source code

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

#define NO_OF_FILES 349         //350 inc of 2 starting at 2; 349 inc of 2 starting at 4; 700 inc of 1 starting at 1
#define END_OF_LOOP 700
#define LINE_BUFFER_SIZE 100

int main()

{
   int i;
   char line[LINE_BUFFER_SIZE] = "";
   char filename[80] = "";
   char *basename = "data_\0";
   char *postfix  = "\0";
   char tempstr[5];
              char ootempstr[80];

   FILE *fh;

              char line1[255];
              FILE* fh1;      //dev file

              char line2[255];
              FILE* fh2;      //max file

              char line3[255];
              FILE* fh3;      //funny val file file

              char linenoiseup[255];
              FILE* fhnup;                        //max height of noise value

              char linenoisedown[255];
              FILE* fhndown;                      //max height of noise value

              char linenormalplat[255];
              FILE* fhnormal;

              double value[5];
               int num = 0;

              double dMultifilesquaredev=0;
              double dLevel=0.2;
              int       iMultifilecounter=0;
              double dMultifilesingledev=0;
              double dMaxmultidev=0;
              double dSinglefiledev=0;
              double dFinaldev;
              char *filename2="multilev.txt";
              char *filename3="maxdev.txt";
              char *filename4="funnyval.txt";

              char *filenamenoiseup="noisemaxup.txt";
              char *filenamenoisedown="noisedown.txt";
              double dGetnoiseup=0;
              double dGetnoisedown=400;

              char *filenamenormalizedpeak="normalizedpeak.txt";
              double dNormalplat=0;
              double dOvershootpercent;

              fh1 = fopen( filename2, "w" );
              fclose( fh1 );

              fhnup = fopen( filenamenoiseup, "w" );
```

```
        fclose( fhnup );

        fhndown = fopen( filenamenoisedown, "w" );
        fclose( fhndown );


        fhnormal = fopen ( filenamenormalizedpeak, "w" );
        fclose( fhnormal );

for( i = 4; i <= END_OF_LOOP; i+=2 )
{
        if (i > END_OF_LOOP)
        printf("problems!!!");

  strcpy( filename, basename );

        strcpy(ootempstr, "");

        if ( i < 100 )
        {
        strcpy(ootempstr, "0");
        }
        if ( i < 10 )
        {
        strcpy(ootempstr, "00");
        }
        if ( i < 1 )
        {
        strcpy(ootempstr, "000");
        }

        strcat( filename, ootempstr );
   sprintf( tempstr, "%i" , i*10 );
        strcat( filename, tempstr );
   strcat( filename, postfix );
   //printf("filename = %s\n", filename);

   fh = fopen( filename, "r" );

    if( fh == NULL )
   {
      printf("Error opening file to be read in \n");
        exit(0);
  }
  else
  {
  fgets( line, LINE_BUFFER_SIZE, fh );    // Get rid of first line

    fgets( line, LINE_BUFFER_SIZE, fh );

        while ( ( line != NULL ) && (strlen(line) > 5) )
     {
       iMultifilecounter++;
       num = atoi( strtok( line, " \t\r\n," ) );
       value[0] = atof( strtok( NULL, " \t\r\n,") );
       value[1] = atof( strtok( NULL, " \t\r\n,") );
       value[2] = atof( strtok( NULL, " \t\r\n,") );
       value[3] = atof( strtok( NULL, " \t\r\n,") );
          //value[4] = atof( strtok( NULL, " \t\r\n,") );

       // Calculations
                                               dMultifilesquaredev=dMultifilesquaredev+pow((value[3]-dLevel),2);

        if( dGetnoiseup <= value[3] )
        {
        dGetnoiseup = value[3];
        }
        if( dGetnoisedown >= value[3] )
        {
        dGetnoisedown = value[3];
        }
        if( i == 400 )
        {
        dNormalplat = value[3];
        }
// End of calculations

               fgets( line, LINE_BUFFER_SIZE, fh );
            }//end while loop reading file


            //printf( "EOF, num = %i\n", num );
        }//end if else for multifilw


                 dMultifilesingledev=(dMultifilesquaredev)/iMultifilecounter;
                 if( dMaxmultidev <= dMultifilesingledev )
                 dMaxmultidev = dMultifilesingledev;
                 }
                 //begin write level dev to file
                 fh1 = fopen( filename2, "a" );
        if( fh1 == NULL )
        {
        printf("Error opening file\n");
        exit(0);
        }
        sprintf( line1, "%f\n",dMultifilesingledev );
                 fputs( line1, fh1 );
                 fclose( fh1 );                                                                  //end write level dev to file
                 //begin writing to noise file up
                 fhnup = fopen( filenamenoiseup, "a" );
                 sprintf( linenoiseup, "%f\n", dGetnoiseup );
                 fputs( linenoiseup,fhnup    );
                 fclose( fhnup );
                 //end writing to noise file up

//begin writing to noise file down
fhndown = fopen( filenamenoisedown, "a" );
sprintf( linenoisedown, "%f\n", dGetnoisedown );
fputs( linenoisedown,fhndown );
fclose( fhndown );
//end writing to noise file down

//reset loop var
dGetnoiseup=0;
dGetnoisedown=400;
//end reset loop var

                                   dSinglefiledev=dSinglefiledev+dMultifilesingledev;
                 fclose(fh);  //close multifile
    }// end of for loop going throught the files

                 //Funny val start

                 dFinaldev=dSinglefiledev/NO_OF_FILES;
fh3 = fopen( filename4, "w" );
                 if( fh3 == NULL )
                 {
                 printf("Error opening file\n");
                 exit(0);
                 }
                 sprintf( line3, "%f", dFinaldev );
                 fputs( line3, fh3 );
                 fclose( fh3 );
                 //printf( "dFinaldev = %d\n", dFinaldev );

                 //Funny val end

                 //begin write max dev to file

                 fh2 = fopen( filename3, "w" );
                 if( fh2 == NULL )
                 {
                 printf("Error opening file\n");
                 exit(0);
                 }
                 sprintf( line2, "%f" , dMaxmultidev );
                 printf( line2,dMaxmultidev );
                 fputs( line2, fh2 );
                 fclose( fh2 );
```

University of Pretoria etd – Kingsley, T C (2005)

```
        //end write max dev to file

        //begin write normalized overshoot to file
        dOvershootpercent=((dMaxmultidev-dNormalplat)/(dNormalplat))*100;

        fhnormal = fopen ( filenamenormalizedpeak, "w" );
        if( fhnormal == NULL )
        {
        printf("Error opening file\n");
        exit(0);
        }
        sprintf( linenormalplat, "%f" , dOvershootpercent );
        fputs( linenormalplat , fhnormal);d
        fclose( fhnormal );

        //end writing normalized overshoot to file


}// end of main
```

APPENDIX J: Fluent UDF momentum source input file

```
#include "udf.h"
#define time 2
#define velocity 30

double AlnrX;

DEFINE_ADJUST(Accl, domain)
{
  double t;

  t = RP_Get_Real("flow-time");
  /*t=0.2;*/

  if (t<=time) {
    AlnrX = velocity/(3.6*time);
  }
  else {
    AlnrX = 0.0;
  }
  printf("time = %f\n",t);
  printf("accel = %f\n",AlnrX);

}


DEFINE_SOURCE(xmom_source, cell, thread, dS, eqn)
{
  double pos[ND_ND];
  double rho;
  double source;

  C_CENTROID(pos,cell,thread);

  rho  = C_R(cell,thread);

  dS[eqn] = 0.0;
  source = AlnrX *rho;

  return source;
}
```

University of Pretoria etd – Kingsley, T C (2005)

APPENDIX K: 2D TDV extraction source code

```
// fluentc.c edited by Thomas Kingsley on 5/11/2002 for
// adaptation to 2-D sloshing and TDV

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

#define NO_OF_FILES 125         // Number of data files to proccess
#define END_OF_LOOP 500                      // Final data file name
#define LINE_BUFFER_SIZE 100

int main()

{
    int i;
    char line[LINE_BUFFER_SIZE] = "";
    char filename[80] = "";
    char *basename = "data_\0";
    char *postfix  = "\0";
    char tempstr[5];
        char ootempstr[80];

    FILE *fh;

        char line1[255];
        FILE* fh1;      //dev file

        char line2[255];
        FILE* fh2;      //max file

        char line3[255];
        FILE* fh3;      //funny val file file

        char linenoiseup[255];
        FILE* fhnup;          //max height of noise value

        char linenoisedown[255];
        FILE* fhndown;        //max height of noise value

        char linenormalplat[255];
        FILE* fhnormal;

        double value[5];
        int num = 0;

        double dMultifilesquaredev=0;
        double dLevel=0.28;
        int        iMultifilecounter=0;
        double dMultifilesingledev=0;
        double dMaxmultidev=0;
        double dSinglefiledev=0;
        double dFinaldev=0;
        char *filename2="multilev.txt";
        char *filename3="maxdev.txt";
        char *filename4="TotalDev.txt";

        char *filenamenoiseup   = "noisemaxup.txt";
        char *filenamenoisedown = "noisedown.txt" ;
        double dGetnoiseup=0;
        double dGetnoisedown=400;

        char *filenamenormalizedpeak="normwave.txt";
        double dNormalplat=0;
        double dOvershootpercent;

char temp1[255];
char temp2[255];

        fh1 = fopen( filename2, "w" );
```

University of Pretoria etd – Kingsley, T C (2005)

```
        fclose( fh1 );
/*
        fhnup = fopen( filenamenoiseup, "w" );
        fclose( fhnup );

        fhndown = fopen( filenamenoisedown, "w" );
        fclose( fhndown );
*/

        fhnormal = fopen ( filenamenormalizedpeak, "w" );
        fclose( fhnormal );

    for( i = 4; i <= END_OF_LOOP; i+=4 )
    {
               if (i > END_OF_LOOP)
                       printf("problems!!!");

        strcpy( filename, basename );

               strcpy(ootempstr, "");

               if ( i < 100 )
               {
                       strcpy(ootempstr, "0");
               }
               if ( i < 10 )
               {
                       strcpy(ootempstr, "00");
               }
               if ( i < 1 )
               {
                       strcpy(ootempstr, "000");
               }

               strcat(  filename, ootempstr );
        sprintf( tempstr , "%i" , i*10 );
               strcat(  filename, tempstr );
        strcat(  filename, postfix );

        fh = fopen( filename, "r" );

        if( fh == NULL )
        {
            printf("Error opening file to be read in \n");
                       exit(0);
        }
        else
        {
            fgets( line, LINE_BUFFER_SIZE, fh );    // Get rid of first line

            fgets( line, LINE_BUFFER_SIZE, fh );

                       while ( ( line != NULL ) && (strlen(line) > 5) )
            {
                iMultifilecounter++;
                              num = atoi( strtok( line, " \t\r\n," ) );
                value[0] = 0;//atof( strtok( NULL, " \t\r\n,") );
                value[1] = atof( strtok( NULL, " \t\r\n,") );
                value[2] = atof( strtok( NULL, " \t\r\n,") );
                value[3] = atof( strtok( NULL, " \t\r\n,") );
                              //value[4] = atof( strtok( NULL, " \t\r\n,") );

                // Calculations

        //dMultifilesquaredev=dMultifilesquaredev+pow((value[3]-dLevel),2);

        dMultifilesquaredev=dMultifilesquaredev+pow(((value[3]-dLevel)*(value[3]-
dLevel)),0.5);

                                      if( dGetnoiseup <= value[3] )
                                      {
                                             dGetnoiseup = value[3];
                                      }
                                      if( dGetnoisedown >= value[3] )
                                      {
                                             dGetnoisedown = value[3];
                                      }
```

University of Pretoria etd – Kingsley, T C (2005)

```
                          // End of calculations

                 fgets( line, LINE_BUFFER_SIZE, fh );
            }//end while loop reading file


            //printf( "EOF, num = %i\n", num );
        }//end if else for multifilw



               dMultifilesingledev=(dMultifilesquaredev)/iMultifilecounter;

               if( i == 400 )
               {
                      dNormalplat = dMultifilesingledev ;
               }

               if( dMaxmultidev <= dMultifilesingledev )
               {
                      dMaxmultidev = dMultifilesingledev;
               }

               //begin write level dev to file

               fh1 = fopen( filename2, "a" );
        if( fh1 == NULL )
        {
            printf("Error opening file\n");
                      exit(0);
        }
               sprintf( line1, "%f\n",dMultifilesingledev );
               fputs( line1, fh1 );
               fclose( fh1 );
               //end write level dev to file

               /*//begin writing to noise file up
               fhnup = fopen( filenamenoiseup, "a" );
               sprintf( linenoiseup, "%f\n", dGetnoiseup );
               fputs( linenoiseup,fhnup   );
               fclose( fhnup );
               //end writing to noise file up

               //begin writing to noise file down
               fhndown = fopen( filenamenoisedown, "a" );
               sprintf( linenoisedown, "%f\n", dGetnoisedown );
               fputs( linenoisedown,fhndown );
               fclose( fhndown );
               //end writing to noise file down*/

               //reset loop var
               dGetnoiseup=0;
               dGetnoisedown=400;
               //end reset loop var

               dSinglefiledev=dSinglefiledev+dMultifilesingledev;
        fclose(fh);  //close multifile
    }// end of for loop going through the files

        // Final deveation start

        dFinaldev=dSinglefiledev/NO_OF_FILES;

        fh3 = fopen( filename4, "w" );
    if( fh3 == NULL )
    {
               printf("Error opening file\n");
               exit(0);
        }
        sprintf( line3, "%f", dFinaldev );
        fputs( line3, fh3 );
        fclose( fh3 );
        //printf( "dFinaldev = %d\n", dFinaldev );

        // Final deveation end
        //begin write max dev to file

        fh2 = fopen( filename3, "w" );
        if( fh2 == NULL )
        {
                printf("Error opening file\n");
                exit(0);
        }
        sprintf( line2, "%f" , dMaxmultidev );
//      printf( line2,dMaxmultidev );
        fputs( line2, fh2 );
        fclose( fh2 );

        //end write max dev to file

        //begin writing percent overshoot of the squared data to file
        dOvershootpercent=((dMaxmultidev-dNormalplat)/(dNormalplat))*100;

        fhnormal = fopen ( filenamenormalizedpeak, "a" );
        if( fhnormal == NULL )
        {
                printf("Error opening file\n");
                exit(0);
        }

//      tempory checks
//      sprintf( temp1, "%f\n" , dNormalplat );
//      printf( temp1, dNormalplat );
//      sprintf( temp2, "%f\n" , dOvershootpercent );
//      printf( temp2, dOvershootpercent);
//
        sprintf( linenormalplat, "%f" , dOvershootpercent );
        fputs( linenormalplat , fhnormal);
        fclose( fhnormal );

        //end writing percent overshoot of the squared data to file

}// end of main
```

University of Pretoria etd – Kingsley, T C (2005)

APPENDIX L: LS-OPT command file – Linear RSM optimisation (Design 1)

```
"2D sloshing optimisation for total deviation value (TDV) with variables of baffle
height, hole size, and baffle centroid location"
Author "Thomas Kingsley, Ken Craig"
$ Created on Mon Sep  1 16:30:29 2003
solvers 1
responses 4
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 3
 Variable 'centroid' 100
  Lower bound variable 'centroid' 10
  Upper bound variable 'centroid' 190
 Variable 'b_height' 100
  Lower bound variable 'b_height' 20
  Upper bound variable 'b_height' 180
 Variable 'hole' 50
  Lower bound variable 'hole' 10
  Upper bound variable 'hole' 140

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "fluent"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "/home/thomas/2dslosh/LSOPT/fluent_script"
  solver input file "/home/thomas/2dslosh/LSOPT/fluentjou2D.jou"
  prepro own
  prepro command "gambit -inp"
  prepro input file "/home/thomas/2dslosh/LSOPT/2D_70p_gambit.jou"
  solver order linear
  solver experiment design dopt
  solver number experiments 7
  solver basis experiment 3toK
  solver concurrent jobs 1
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'TDV' 1000 0 "cat TotalDev.txt"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "fluent"
$
 response 'too_high_baf' expression {(190-0.5*b_height)-centroid}
 response 'too_low_baf' expression {(10+0.5*b_height)-centroid}
 response 'too_big_hole' expression {(0.8*b_height)-hole}

$
$ OBJECTIVE FUNCTIONS
$
 objectives 1
 objective 'TDV' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 3
  move
 constraint 'too_high_baf'
  lower bound constraint 'too_high_baf' 0
```

University of Pretoria etd – Kingsley, T C (2005)

```
 constraint 'too_low_baf'
  upper bound constraint 'too_low_baf' 0
 constraint 'too_big_hole'
  lower bound constraint 'too_big_hole' 0
$
$ JOB INFO
$
 iterate param design 0.001
 iterate param objective 0.001
 iterate param stoppingtype and
 iterate 10
STOP
```

University of Pretoria etd – Kingsley, T C (2005)

APPENDIX M: LSOPT command file – Quadratic RSM optimisation (Design 1)

```
"2D sloshing optimisation for total deviation value (TDV) with variables of baffle
height, hole size, and baffle centroid location"
Author "Thomas Kingsley, Ken Craig"
$ Created on Mon Sep 15 12:00:18 2003
solvers 1
responses 5
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 3
 Variable 'centroid' 100
  Lower bound variable 'centroid' 10
  Upper bound variable 'centroid' 190
 Variable 'b_height' 100
  Lower bound variable 'b_height' 20
  Upper bound variable 'b_height' 180
 Variable 'hole' 50
  Lower bound variable 'hole' 10
  Upper bound variable 'hole' 140

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "fluent"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "/home/thomas/2dslosh/LSOPT/fluent_script"
  solver input file "/home/thomas/2dslosh/LSOPT/fluentjou2D.jou"
  prepro own
  prepro command "gambit -inp"
  prepro input file "/home/thomas/2dslosh/LSOPT/2D_70p_gambit.jou"
  solver order quadratic
  solver experiment design dopt
  solver number experiments 16
  solver basis experiment 5toK
  solver concurrent jobs 1
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'TDV' 1000 0 "cat TotalDev.txt"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "fluent"
$
 response 'too_high_baf' expression {(190-0.5*b_height)-centroid}
 response 'too_low_baf' expression {(10+0.5*b_height)-centroid}
 response 'too_big_hole' expression {(0.8*b_height)-hole}
 response 'baffle_length' expression {4*(b_height-hole)}

$
$ OBJECTIVE FUNCTIONS
$
 objectives 1
 objective 'TDV' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 4
  move
 constraint 'too_high_baf'
```

```
  lower bound constraint 'too_high_baf' 0
 constraint 'too_low_baf'
  upper bound constraint 'too_low_baf' 0
 constraint 'too_big_hole'
  lower bound constraint 'too_big_hole' 0
  stay
 constraint 'baffle_length'
  lower bound constraint 'baffle_length' 0
  upper bound constraint 'baffle_length' 800
$
$ JOB INFO
$
 iterate param design 0.001
 iterate param objective 0.001
 iterate param stoppingtype and
 iterate 8
STOP
```

University of Pretoria etd – Kingsley, T C (2005)

APPENDIX N: LSOPT command file – Neural Network optimisation (Design 1)

```
"2D sloshing optimisation for total deviation value (TDV) with variables of baffle
height, hole size, and baffle centroid location"
Author "Thomas Kingsley, Ken Craig"
$ Created on Mon Sep  8 10:05:28 2003
solvers 1
responses 4
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 3
 Variable 'centroid' 100
  Lower bound variable 'centroid' 10
  Upper bound variable 'centroid' 190
 Variable 'b_height' 100
  Lower bound variable 'b_height' 20
  Upper bound variable 'b_height' 180
 Variable 'hole' 50
  Lower bound variable 'hole' 10
  Upper bound variable 'hole' 140

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "fluent"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "/home/thomas/2dslosh/LSOPT/fluent_script"
  solver input file "/home/thomas/2dslosh/LSOPT/fluentjou2D.jou"
  prepro own
  prepro command "gambit -inp"
  prepro input file "/home/thomas/2dslosh/LSOPT/2D_70p_gambit.jou"
  solver order FF
  solver update doe
  solver experiment design latin_hypercube
  solver number experiments 129
  solver concurrent jobs 1
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'TDV' 1000 0 "cat TotalDev.txt"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "fluent"
$
 response 'too_high_baf' expression {(190-0.5*b_height)-centroid}
 response 'too_low_baf' expression {(10+0.5*b_height)-centroid}
 response 'too_big_hole' expression {(0.8*b_height)-hole}

$
$ OBJECTIVE FUNCTIONS
$
 objectives 1
 objective 'TDV' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 3
  move
 constraint 'too_high_baf'
```

```
  strict
  lower bound constraint 'too_high_baf' 0
  slack
 constraint 'too_low_baf'
  strict
  upper bound constraint 'too_low_baf' 0
 constraint 'too_big_hole'
  lower bound constraint 'too_big_hole' 0
  slack
$
$ JOB INFO
$
 iterate param design 0.001
 iterate param objective 0.001
 iterate param stoppingtype and
 iterate 1
STOP
```

APPENDIX O: LSOPT command file – Quadratic RSM optimisation (Design 2)

```
"2D sloshing optimisation for total deviation value (TDV) with 4 variables and
Variable Fill"
Author "Thomas Kingsley, Ken Craig"
$ Created on Wed Sep 17 11:02:55 2003
solvers 1
responses 6
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 4
 Variable 'mid_baf_centroid' 100
  Lower bound variable 'mid_baf_centroid' 15
  Upper bound variable 'mid_baf_centroid' 185
 Variable 'side_baf_centroid' 100
  Lower bound variable 'side_baf_centroid' 15
  Upper bound variable 'side_baf_centroid' 185
 Variable 'mid_baf_height' 100
  Lower bound variable 'mid_baf_height' 10
  Upper bound variable 'mid_baf_height' 180
 Variable 'side_baf_width' 100
  Lower bound variable 'side_baf_width' 10
  Upper bound variable 'side_baf_width' 180
$
$ CONSTANTS
$
constants 1
 Constant 'fill_level' 0.14

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "fluent"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "../../fluent_script"
  solver input file "fluentjou2D.jou"
  prepro own
  prepro command "gambit -inp"
  prepro input file "2D_VF_gambit.jou"
  solver order quadratic
  solver experiment design dopt
  solver number experiments 23
  solver basis experiment 5toK
  solver concurrent jobs 1
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'TDV' 1000 0 "cat TotalDev.txt"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "fluent"
$
 response 'SB_side_constraint' expression {side_baf_centroid-side_baf_width/2}
 response 'SB_mid_constaint' expression {side_baf_centroid+side_baf_width/2}
 response 'MB_up_constaint' expression {mid_baf_centroid+mid_baf_height/2}
 response 'MB_low_constraint' expression {mid_baf_centroid-mid_baf_height/2}
 response 'baffle_length' expression {mid_baf_height+2*side_baf_width}

$
```

```
$ OBJECTIVE FUNCTIONS
$
 objectives 1
 objective 'TDV' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 5
  move
 constraint 'SB_side_constraint'
  strict
  lower bound constraint 'SB_side_constraint' 10
  slack
 constraint 'SB_mid_constaint'
  strict
  upper bound constraint 'SB_mid_constaint' 190
 constraint 'MB_up_constaint'
  slack
  strict
  upper bound constraint 'MB_up_constaint' 190
 constraint 'MB_low_constraint'
  lower bound constraint 'MB_low_constraint' 10
  slack
  stay
 constraint 'baffle_length'
  lower bound constraint 'baffle_length' 0
  upper bound constraint 'baffle_length' 540
$
$ JOB INFO
$
 iterate param design 0.001
 iterate param objective 0.001
 iterate param stoppingtype and
 iterate 8
STOP
```

APPENDIX P: LSOPT command file – Quadratic RSM optimisation (Design 2B)

```
"2dslosh opt (full size) (design 2B"
Author "Thomas Kingsley, Ken Craig"
$ Created on Tue Dec  9 12:27:54 2003
solvers 1
responses 6
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 4
 Variable 'mid_baf_centroid' 100
  Lower bound variable 'mid_baf_centroid' 60
  Upper bound variable 'mid_baf_centroid' 320
 Variable 'side_baf_centroid' 100
  Lower bound variable 'side_baf_centroid' 15
  Upper bound variable 'side_baf_centroid' 185
 Variable 'mid_baf_height' 100
  Lower bound variable 'mid_baf_height' 40
  Upper bound variable 'mid_baf_height' 320
 Variable 'side_baf_width' 100
  Lower bound variable 'side_baf_width' 10
  Upper bound variable 'side_baf_width' 180
$
$ CONSTANTS
$
constants 1
 Constant 'fill_level' 0.28

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "fluent"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "../../fluent_script"
  solver input file "fluentjou2D.jou"
  prepro own
  prepro command "gambit -inp"
  prepro input file "2D_tom_gambit.jou"
  solver order linear
  solver experiment design dopt
  solver number experiments 8
  solver basis experiment 3toK
  solver concurrent jobs 1
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'TDV' 1000 0 "cat TotalDev.txt"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "fluent"
$
 response 'SB_side_constraint' expression {side_baf_centroid-side_baf_width/2}
 response 'SB_mid_constaint' expression {side_baf_centroid+side_baf_width/2}
 response 'MB_up_constaint' expression {mid_baf_centroid+mid_baf_height/2}
 response 'MB_low_constaint' expression {mid_baf_centroid-mid_baf_height/2}
 response 'baffle_length' expression {mid_baf_height+2*side_baf_width}

$
$ OBJECTIVE FUNCTIONS
$
```

```
 objectives 1
 objective 'TDV' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 5
  move
 constraint 'SB_side_constraint'
  strict
  lower bound constraint 'SB_side_constraint' 10
  slack
 constraint 'SB_mid_constaint'
  strict
  upper bound constraint 'SB_mid_constaint' 190
 constraint 'MB_up_constaint'
  slack
  strict
  upper bound constraint 'MB_up_constaint' 340
 constraint 'MB_low_constraint'
  lower bound constraint 'MB_low_constraint' 40
  slack
  stay
 constraint 'baffle_length'
  lower bound constraint 'baffle_length' 0
  upper bound constraint 'baffle_length' 540
$
$ JOB INFO
$
 iterate param design 0.001
 iterate param objective 0.001
 iterate param stoppingtype and
 iterate 6
STOP
```

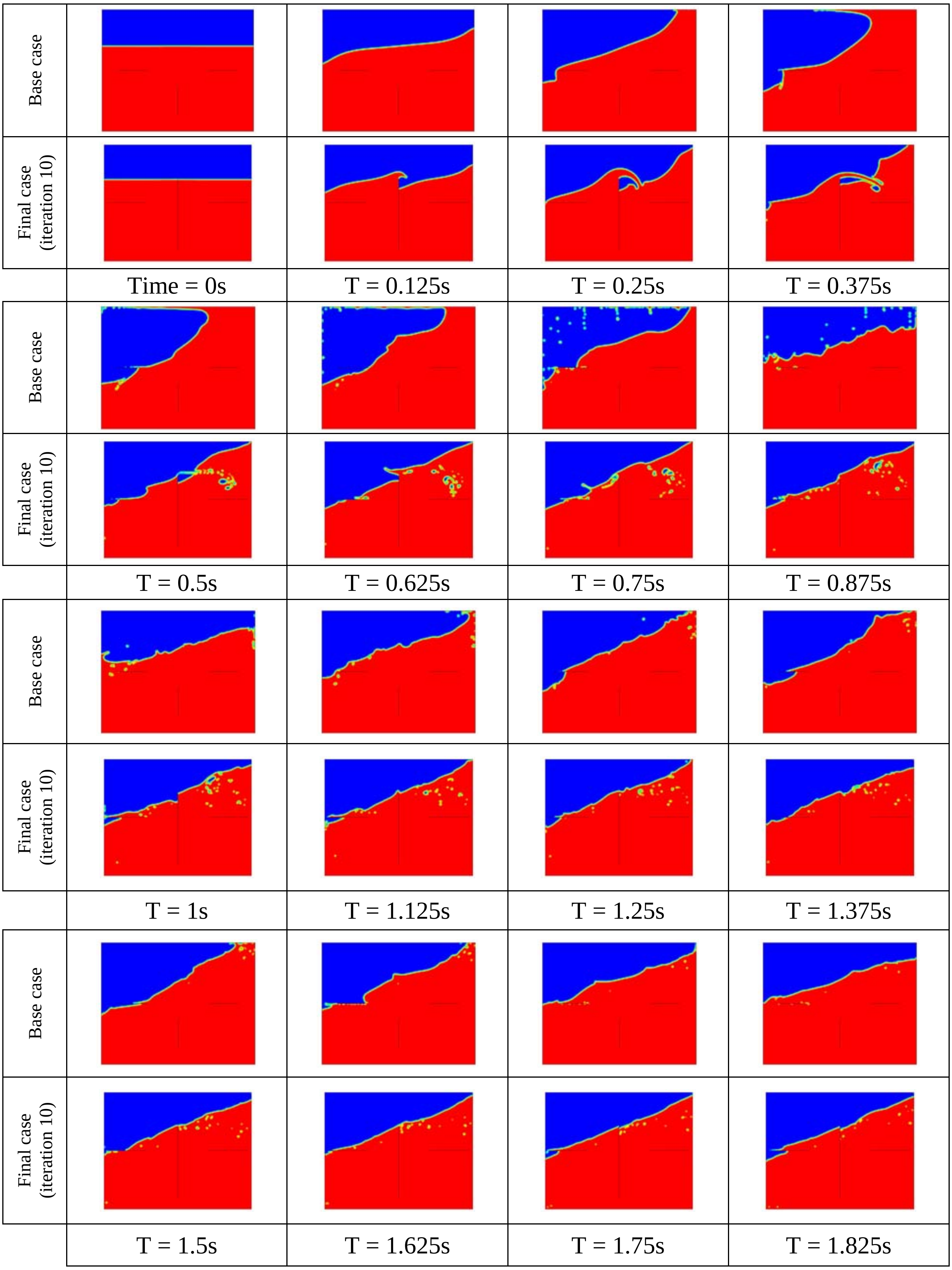
APPENDIX Q: LSOPT command file – Quadratic RSM saddle-point optimisation (Design 2)

```
"2D sloshing optimisation for total deviation value (TDV) with 4 variables and
Variable Fill"
Author "Thomas Kingsley, Ken Craig"
$ Created on Wed Oct  8 12:55:19 2003
solvers 1
responses 7
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 5
 Variable 'fill_level' 0.1
  Lower bound variable 'fill_level' 0.01
  Upper bound variable 'fill_level' 0.19
  Variable 'fill_level' max
 Variable 'mid_baf_centroid' 100
  Lower bound variable 'mid_baf_centroid' 15
  Upper bound variable 'mid_baf_centroid' 185
 Variable 'side_baf_centroid' 100
  Lower bound variable 'side_baf_centroid' 15
  Upper bound variable 'side_baf_centroid' 185
 Variable 'mid_baf_height' 100
  Lower bound variable 'mid_baf_height' 10
  Upper bound variable 'mid_baf_height' 180
 Variable 'side_baf_width' 100
  Lower bound variable 'side_baf_width' 10
  Upper bound variable 'side_baf_width' 180

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "fluent"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "../../fluent_script"
  solver input file "fluentjou2D.jou"
  prepro own
  prepro command "gambit -inp"
  prepro input file "2D_VF_gambit.jou"
  solver order quadratic
  solver experiment design dopt
  solver number experiments 32
  solver basis experiment 5toK
  solver concurrent jobs 1
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'TDV' 1000 0 "cat TotalDev.txt"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "fluent"
$
 response 'SB_side_constraint' expression {side_baf_centroid-side_baf_width/2}
 response 'SB_mid_constaint' expression {side_baf_centroid+side_baf_width/2}
 response 'MB_up_constaint' expression {mid_baf_centroid+mid_baf_height/2}
 response 'MB_low_constraint' expression {mid_baf_centroid-mid_baf_height/2}
 response 'baffle_length' expression {mid_baf_height+2*side_baf_width}
 response 'Percent_Fill' expression {100*fill_level/0.2}

$
$ OBJECTIVE FUNCTIONS
```

University of Pretoria etd – Kingsley, T C (2005)

```
$
 objectives 1
 objective 'TDV' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 6
  move
 constraint 'SB_side_constraint'
  strict
  lower bound constraint 'SB_side_constraint' 10
  slack
 constraint 'SB_mid_constaint'
  strict
  upper bound constraint 'SB_mid_constaint' 190
 constraint 'MB_up_constaint'
  slack
  strict
  upper bound constraint 'MB_up_constaint' 190
 constraint 'MB_low_constraint'
  lower bound constraint 'MB_low_constraint' 10
  slack
  stay
 constraint 'baffle_length'
  lower bound constraint 'baffle_length' 0
  upper bound constraint 'baffle_length' 540
 constraint 'Percent_Fill'
  lower bound constraint 'Percent_Fill' 10
  upper bound constraint 'Percent_Fill' 90
$
$ JOB INFO
$
 iterate param design 0.001
 iterate param objective 0.001
 iterate param stoppingtype and
 iterate 5
STOP
```

APPENDIX R: Comparative sloshing frames for design 2b (Case 7 of Chapter 4)



APPENDIX S: Sample Patran baffled tank session file

```
$
$               Patran session file for generating 90% full tank
$                   2D extended to 3D geometry ; baffles 1
$                   Created by: Thomas Kingsley 17/11/2003
$
uil_file_new.go( "/usr/local/msc/patran2003r2/lsdyna_prefrence.db",
"patran_tester1.db" )
db_set_pref( 303, 3, 0, FALSE, 0.0005, "" )
$
$
$                             Define Variables
$
$
REAL baffle_h
REAL baf_cent
REAL fe_size1
REAL fe_size2
$
$
$                             Variable values
$
$
baffle_h = <<baffle_H>>
baf_cent = <<baffle_centroid>>
baffle_t = 0.0001
$
fe_size1 = 0.01
fe_size2 = 0.0025
$
$                             Front
$
$
gu_fit_view(  )
ga_view_aa_set( -90., 0., 90. )
sys_poll_option( 2 )
ga_group_create( "rigid" )
ga_viewport_group_post( "", "rigid" )
sys_poll_option( 0 )
ga_group_current_set( "rigid" )
STRING asm_create_grid_xyz_created_ids[VIRTUAL]
asm_const_grid_xyz( "1", "[0 0 0]", "Coord 0", asm_create_grid_xyz_created_ids )
STRING asm_create_grid_xyz_created_ids[VIRTUAL]
asm_const_grid_xyz( "2", "[0 0 `baf_cent`]", "Coord 0",
asm_create_grid_xyz_created_ids )
STRING sgm_transform_point_created_ids[VIRTUAL]
asm_transform_grid_translate( "3", "<0 0 `-baffle_h/2`>", "Coord 0", 1, FALSE, FALSE,
"Point 2", sgm_transform_point_created_ids )
asm_transform_grid_translate( "4", "<0 0 `baffle_h/2`>", "Coord 0", 1, FALSE, FALSE,
"Point 2", sgm_transform_point_created_ids )
STRING asm_create_grid_xyz_created_ids[VIRTUAL]
asm_const_grid_xyz( "5", "[0 0 0.4]", "Coord 0", asm_create_grid_xyz_created_ids )
STRING asm_create_grid_xyz_created_ids[VIRTUAL]
asm_const_grid_xyz( "6", "[0 0 0.36]", "Coord 0", asm_create_grid_xyz_created_ids )
STRING asm_line_2point_created_ids[VIRTUAL]
asm_const_line_2point( "1", "Point 1", "Point 3", 0, "", 50., 1,
asm_line_2point_created_ids )
asm_const_line_2point( "2", "Point 3", "Point 4", 0, "", 50., 1,
asm_line_2point_created_ids )
asm_const_line_2point( "3", "Point 4", "Point 6", 0, "", 50., 1,
asm_line_2point_created_ids )
asm_const_line_2point( "4", "Point 6", "Point 5", 0, "", 50., 1,
asm_line_2point_created_ids )
STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_translate( "5", "curve", "<0 0.4 0>", "Coord 0", 1, FALSE, "Curve 1:4",
sgm_transform_curve_created_ids )
STRING sgm_surface_2curve_created_ids[VIRTUAL]
sgm_const_surface_2curve( "1", "Curve 1", "Curve 5", sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "2", "Curve 2", "Curve 6", sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "3", "Curve 3", "Curve 7", sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "4", "Curve 4", "Curve 8", sgm_surface_2curve_created_ids )
$
```

```
$                              shells
$
INTEGER fem_create_mesh_surfa_num_nodes
INTEGER fem_create_mesh_surfa_num_elems
STRING fem_create_mesh_s_nodes_created[VIRTUAL]
STRING fem_create_mesh_s_elems_created[VIRTUAL]
fem_create_mesh_surf_4( "IsoMesh", 49152, "Surface 2", 1, ["`fe_size1`"], "Quad4",
"1", "1", "Coord 0", "Coord 0", fem_create_mesh_surfa_num_nodes,
fem_create_mesh_surfa_num_elems, fem_create_mesh_s_nodes_created,
fem_create_mesh_s_elems_created )
fem_create_mesh_surf_4( "IsoMesh", 49152, "Surface 1 3", 1, ["`fe_size1`"], "Quad4",
"5000", "5000", "Coord 0", "Coord 0", fem_create_mesh_surfa_num_nodes,
fem_create_mesh_surfa_num_elems, fem_create_mesh_s_nodes_created,
fem_create_mesh_s_elems_created )
fem_create_mesh_surf_4( "IsoMesh", 49152, "Surface 4", 1, ["`fe_size1`"], "Quad4",
"10000", "10000", "Coord 0", "Coord 0", fem_create_mesh_surfa_num_nodes,
fem_create_mesh_surfa_num_elems, fem_create_mesh_s_nodes_created,
fem_create_mesh_s_elems_created )
$
$
$                                Baffles
$
$
sys_poll_option( 2 )
ga_group_create( "baffles" )
ga_group_current_set( "baffles" )
STRING fem_transform_elem_created_nids[VIRTUAL]
STRING fem_transform_elem_created_eids[VIRTUAL]
STRING fem_transform_elem_deleted_nids[VIRTUAL]
STRING fem_transform_elem_deleted_eids[VIRTUAL]
fem_translate_elems_1( "20000", "<0.25 0 0>", "Coord 0", 1, FALSE, 2, "Elm 1:4500",
fem_transform_elem_created_nids, fem_transform_elem_created_eids,  @
fem_transform_elem_deleted_nids, fem_transform_elem_deleted_eids )
$? YESFORALL 2009007
$
$                              Water
$
sys_poll_option( 2 )
ga_group_create( "water" )
ga_group_current_set( "water" )
$
INTEGER fem_sweep_elems_n_nodes_created
INTEGER fem_sweep_elems_n_elems_created
STRING fem_sweep_elems_ex_created_nids[VIRTUAL]
STRING fem_sweep_elems_ex_created_eids[VIRTUAL]
fem_sweep_extrude_1( "30000", "30000", "Coord 0", "<0.25 0 0>", "0.25", "0.0 ",  @
2, "Elm 1:9500", 10, ["Bar2", "Quad4", "Quad8", "Quad12", "Wedge6", "Wedge15" @
, "Wedge24", "Hex8", "Hex20", "Hex32", "", "", "", "", "", "", "", "", "", "", @
 "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "",  @
"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "" @
, "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "",  @
"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "" @
, "", ""], "Coord 0", "Coord 0", "Uniform: Element Length", ["3", "1.5", "0.1" @
, "0.2", "`fe_size1`", "2", "", "", "", ""], fem_sweep_elems_n_nodes_created,  @
fem_sweep_elems_n_elems_created, fem_sweep_elems_ex_created_nids,  @
fem_sweep_elems_ex_created_eids )
$? YESFORALL 2001200
$
STRING fem_transform_elem_created_nids[VIRTUAL]
STRING fem_transform_elem_created_eids[VIRTUAL]
STRING fem_transform_elem_deleted_nids[VIRTUAL]
STRING fem_transform_elem_deleted_eids[VIRTUAL]
fem_translate_elems_1( "100000", "<0.25 0 0>", "Coord 0", 1, FALSE, 2,  @
"Elm 30000:95000", fem_transform_elem_created_nids,  @
fem_transform_elem_created_eids, fem_transform_elem_deleted_nids,  @
fem_transform_elem_deleted_eids )
$? YESFORALL 2009007
$
$                              Air
$
sys_poll_option( 2 )
ga_group_create( "air" )
ga_group_current_set( "air" )
$
INTEGER fem_sweep_elems_n_nodes_created
INTEGER fem_sweep_elems_n_elems_created
STRING fem_sweep_elems_ex_created_nids[VIRTUAL]
STRING fem_sweep_elems_ex_created_eids[VIRTUAL]
fem_sweep_extrude_1( "300000", "300000", "Coord 0", "<0.25 0 0>", "0.25", "0.0 " @
, 2, "Elm 10000:19000", 10, ["Bar2", "Quad4", "Quad8", "Quad12", "Wedge6",  @
"Wedge15", "Wedge24", "Hex8", "Hex20", "Hex32", "", "", "", "", "", "", "", "" @
, "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "",  @
"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "" @
, "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "",  @
"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "" @
, "", "", "", ""], "Coord 0", "Coord 0", "Uniform: Element Length", ["3",  @
"1.5", "0.1", "0.2", "`fe_size1`", "2", "", "", "", ""],  @
fem_sweep_elems_n_nodes_created, fem_sweep_elems_n_elems_created,  @
fem_sweep_elems_ex_created_nids, fem_sweep_elems_ex_created_eids )
$? YESFORALL 2001200
$
fem_translate_elems_1( "320000", "<0.25 0 0>", "Coord 0", 1, FALSE, 2,  @
"Elm 300000:318000", fem_transform_elem_created_nids,  @
fem_transform_elem_created_eids, fem_transform_elem_deleted_nids,  @
fem_transform_elem_deleted_eids )
$? YESFORALL 2009007
$
$                              Close box
$
ga_group_current_set( "rigid" )
STRING sgm_transform_surf__created_ids[VIRTUAL]
sgm_transform_translate( "5", "surface", "<0.25 0 0>", "Coord 0", 2, FALSE,  @
"Surface 1:4", sgm_transform_surf__created_ids )
STRING sgm_surface_2curve_created_ids[VIRTUAL]
sgm_const_surface_2curve( "13", "Curve 1", "Surface 5.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "14", "Curve 2", "Surface 6.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "15", "Curve 3", "Surface 7.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "16", "Curve 4", "Surface 8.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "17", "Surface 5.4", "Surface 9.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "18", "Surface 6.4", "Surface 10.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "19", "Surface 7.4", "Surface 11.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "20", "Surface 8.4", "Surface 12.4",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "21", "Surface 12.2", "Surface 8.2",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "22", "Surface 8.2", "Curve 8",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "23", "Curve 5", "Surface 5.2",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "24", "Surface 5.2", "Surface 9.2",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "25", "Surface 10.2", "Surface 6.2",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "26", "Curve 6", "Surface 6.2",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "27", "Curve 7", "Surface 7.2",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "28", "Surface 1.1", "Surface 5.1",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "29", "Surface 5.1", "Surface 9.1",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "30", "Surface 4.3", "Surface 8.3",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "31", "Surface 8.3", "Surface 12.3",  @
sgm_surface_2curve_created_ids )
sgm_const_surface_2curve( "32", "Surface 7.2", "Surface 11.2",  @
sgm_surface_2curve_created_ids )
INTEGER fem_create_mesh_surfa_num_nodes
INTEGER fem_create_mesh_surfa_num_elems
STRING fem_create_mesh_s_nodes_created[VIRTUAL]
STRING fem_create_mesh_s_elems_created[VIRTUAL]
fem_create_mesh_surf_4( "IsoMesh", 49152, "Surface 9:32", 1, ["`fe_size1`"],  @
"Quad4", "500000", "500000", "Coord 0", "Coord 0",  @
fem_create_mesh_surfa_num_nodes, fem_create_mesh_surfa_num_elems,  @
fem_create_mesh_s_nodes_created, fem_create_mesh_s_elems_created )
$
$                              Create Materials (dummy)
```

```
$
material.create( "Analysis code ID", 10001, "Analysis type ID", 1, "rigid_mat" @
, 0, "Date: 27-Aug-03           Time: 12:17:54", "Isotropic", 1,  @
"Directionality", 1, "Linearity", 11001, "Homogeneous", 0, "Rigid", 11001,  @
"Model Options & IDs", ["Material Type 20", "", "", "", ""], [11006, 0, 0, 0,  @
0], "Active Flag", 1, "Create", 10, "External Flag", FALSE, "Property IDs", [ @
"Density", "Elastic Modulus", "Poisson Ratio"], [16, 2, 5, 0],  @
"Property Values", ["7280", "2e11", "0.3", ""] )
material.create( "Analysis code ID", 10001, "Analysis type ID", 1,  @
"baffle_mat", 0, "Date: 27-Aug-03           Time: 12:17:54", "Isotropic", 1,  @
"Directionality", 1, "Linearity", 1, "Homogeneous", 0, "Linear Elastic", 1,  @
"Model Options & IDs", ["Linear Elastic(MAT1)", "Solid", "", "", ""], [11002,  @
11003, 0, 0, 0], "Active Flag", 1, "Create", 10, "External Flag", FALSE,  @
"Property IDs", ["Density", "Elastic Modulus", "Poisson Ratio"], [16, 2, 5, 0] @
, "Property Values", ["7280.", "2E+11", "0.30000001", ""] )
material.create( "Analysis code ID", 10001, "Analysis type ID", 1, "water_mat" @
, 0, "Date: 27-Aug-03           Time: 12:17:54", "Isotropic", 1,  @
"Directionality", 1, "Linearity", 1, "Homogeneous", 0, "Linear Elastic", 1,  @
"Model Options & IDs", ["Linear Elastic(MAT1)", "Fluid", "", "", ""], [11002,  @
11004, 0, 0, 0], "Active Flag", 1, "Create", 10, "External Flag", FALSE,  @
"Property IDs", ["Density", "Bulk Modulus"], [16, 14, 0], "Property Values", [ @
"1000.", "2e9", ""] )
material.create( "Analysis code ID", 10001, "Analysis type ID", 1, "air_mat",  @
0, "Date: 27-Aug-03           Time: 12:17:54", "Isotropic", 1,  @
"Directionality", 1, "Linearity", 1, "Homogeneous", 0, "Linear Elastic", 1,  @
"Model Options & IDs", ["Linear Elastic(MAT1)", "Fluid", "", "", ""], [11002,  @
11004, 0, 0, 0], "Active Flag", 1, "Create", 10, "External Flag", FALSE,  @
"Property IDs", ["Density", "Bulk Modulus"], [16, 14, 0], "Property Values", [ @
"1.", "2e5", ""] )
$
$                              Assign materials to elements
$
elementprops_create( "rigid.1", 51, 25, 35, 11003, 1, 20, [13, 20, 36, 1004,  @
11044, 11136, 11026, 11027, 1011, 11182], [5, 2, 1, 1, 4, 3, 4, 4, 1, 1], [ @
"m:rigid_mat", "", "0.01", "", "", "", "", "", "", ""],  @
"Element 1:19000 500000:600000" )
elementprops_create( "baffle.2", 51, 25, 35, 11003, 1, 20, [13, 20, 36, 1004,  @
11044, 11136, 11026, 11027, 1011, 11182], [5, 2, 1, 1, 4, 3, 4, 4, 1, 1], [ @
"m:baffle_mat", "", "0.001", "", "", "", "", "", "", ""],  @
"Element 20000:29000" )
elementprops_create( "air.3", 71, 25, 20, 11027, 1, 20, [13, 20, 1011, 11182], @
 [5, 2, 1, 1], ["m:air_mat", "", " ", " "], "Element 300000:400000" )
elementprops_create( "water.4", 71, 25, 20, 11027, 1, 20, [13, 20, 1011,  @
11182], [5, 2, 1, 1], ["m:water_mat", "", " ", " "], "Element 30000:290000" )
$
$                              Equivilence
$
REAL fem_equiv_all_x_equivtol
INTEGER fem_equiv_all_x_segment
fem_equiv_all_group3( [" "], 0, "", 1, 1.0E-04, FALSE,  @
fem_equiv_all_x_equivtol, fem_equiv_all_x_segment )
$
$                              Delete duplicates
$
INTEGER fem_verify_duplicates_num_dupl
STRING fem_verify_duplicates_keep_ids[VIRTUAL]
STRING fem_verify_duplicates_toss_ids[VIRTUAL]
fem_verify_element_duplicates( TRUE, TRUE, fem_verify_duplicates_num_dupl,
fem_verify_duplicates_keep_ids, fem_verify_duplicates_toss_ids )
$
$                              Translate to LS-Dyna keyword file
$
$
loadsbcs_eval_all(  )
jobfile.open( "patran_keyword", "ANALYZE NO JOBFILE" )
jobfile.create_param( "version_opt", 0, 0., "LS-930", 4 )
jobfile.create_param( "sep_mesh", 0, 0., "OFF", 4 )
jobfile.create_param( "sep_mat", 0, 0., "OFF", 4 )
jobfile.create_param( "label_file", 0, 0., "*.sif", 4 )
jobfile.create_param( "list_file", 0, 0., "", 1 )
jobfile.create_param( "cpu_time_limit", 0, 0., "", 3 )
jobfile.create_param( "termination_step", 0, 0., "", 1 )
jobfile.create_param( "termination_time", 0, 0., "", 3 )
jobfile.create_param( "time_step_ratio_limit", 0, 0., "", 3 )
jobfile.create_param( "energy_ratio_limit", 0, 0., "", 3 )
jobfile.create_param( "mass_ratio_limit", 0, 0., "", 3 )
jobfile.create_param( "time_size_calc_basis", 0, 0., "Area/Edge", 4 )
```

University of Pretoria etd – Kingsley, T C (2005)

```
jobfile.create_param( "initial_time_step", 0, 0., "", 3 )
jobfile.create_param( "step_scale_factor", 0, 0.89999998, "", 3 )
jobfile.create_param( "step_size_for_mass_scale", 0, 0., "", 3 )
jobfile.create_param( "shell_minimum_time_step", 0, 0., "", 3 )
jobfile.create_param( "load_curve_max_time_step", 0, 0., "", 1 )
jobfile.create_param( "erosion_flag", 0, 0., "OFF", 4 )
jobfile.create_param( "mass_scaling_first_step", 0, 0., "OFF", 4 )
jobfile.create_param( "Number_of_cpus", 1, 0., "", 1 )
jobfile.create_param( "One_rhs_only", 0, 0., "OFF", 4 )
jobfile.create_param( "Consistency_flg", 0, 0., "OFF", 4 )
jobfile.create_param( "parallel_force", 0, 0., "OFF", 4 )
jobfile.create_param( "optmenu_relaxation", 0, 0., "None.Active", 4 )
jobfile.create_param( "geometry_file", 0, 0., "", 4 )
jobfile.create_param( "relax_termination_time", 0, 1E+30, "", 3 )
jobfile.create_param( "con_tolerance", 0, 0.001, "", 3 )
jobfile.create_param( "Iterations_checks", 250, 0., "", 1 )
jobfile.create_param( "auto_control", 0, 0., "OFF", 4 )
jobfile.create_param( "papadrakakis", 0, 0.039999999, "", 3 )
jobfile.create_param( "Relaxation_Factor", 0, 0.995, "", 3 )
jobfile.create_param( "Time_scale_Factor", 0, 0.89999998, "", 3 )
jobfile.create_param( "global_damping_curve", 0, 0., "f:ouatiafltlak", 4 )
jobfile.create_param( "system_damping_constant", 0, 0., "", 3 )
jobfile.create_param( "linear_viscosity_coefficient", 0, 0.059999999, "", 3 )
jobfile.create_param( "quadratic_viscosity_coefficient", 0, 1.5, "", 3 )
jobfile.create_param( "hourglass_viscosity_type", 0, 0., "LS_DYNA", 4 )
jobfile.create_param( "hourglass_viscosity_coefficient", 0, 0.1, "", 3 )
jobfile.create_param( "hourglass_energy_calc", 0, 0., "OFF", 4 )
jobfile.create_param( "stonewall_energy_diss", 0, 0., "ON", 4 )
jobfile.create_param( "sliding_int_energy_diss", 0, 0., "OFF", 4 )
jobfile.create_param( "rayleigh_energy_diss", 0, 0., "OFF", 4 )
jobfile.create_param( "Warning_angle_warpage", 0, 20., "", 3 )
jobfile.create_param( "Treat_degen_quads_as_tris", 0, 0., "OFF", 4 )
jobfile.create_param( "shell_theory", 0, 0., "Belytschko", 4 )
jobfile.create_param( "warping_stiffness", 0, 0., "Belytschko-Tsay", 4 )
jobfile.create_param( "normal_update", 0, 0., "Each Cycle", 4 )
jobfile.create_param( "update_option", -1, 0., "", 1 )
jobfile.create_param( "update_shell_thick", 0, 0., "OFF", 4 )
jobfile.create_param( "plastics_method", 0, 0., "Secant", 4 )
jobfile.create_param( "consider_shell_thickness", 0, 0.,  @
"Thickness not considered", 4 )
jobfile.create_param( "check_penetration", 0, 0., "ON", 4 )
jobfile.create_param( "max_check_multiplier", 0, 4., "", 3 )
jobfile.create_param( "step_bet_search", 10, 0., "", 1 )
jobfile.create_param( "search_old_surface", 0, 0., "OFF", 4 )
jobfile.create_param( "stiffness_value", 0, 0., "Min. of Master & Slave", 4 )
jobfile.create_param( "scale_interface", 0, 0.1, "", 3 )
jobfile.create_param( "scale_rigid_wall", 0, 0., "", 3 )
jobfile.create_param( "shell_thick_include", 0, 0., "OFF", 4 )
jobfile.create_param( "auto_reorientation", 0, 0., "Active for Automated", 4 )
jobfile.create_param( "control_subroutine", 0, 0., "", 1 )
jobfile.create_param( "friction_subroutine", 0, 0., "", 1 )
jobfile.create_param( "bin_state_time_int", 0, 0., "", 3 )
jobfile.create_param( "excl_damp_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "local_coord_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "one_plot_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "elim_rigid_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "output_hglass_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "output_time_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "num_beam_int_dbox", 0, 0., "", 1 )
jobfile.create_param( "inc_surf_strain_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "excl_sh_tensor_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "excl_sh_strain_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "excl_sh_res_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "excl_int_energy_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "num_var_sol_dbox", 0, 0., "", 1 )
jobfile.create_param( "num_var_sh_dbox", 0, 0., "", 1 )
jobfile.create_param( "num_sh_int_dbox", 0, 0., "", 1 )
jobfile.create_param( "bin_history_time_int", 0, 0., "", 3 )
jobfile.create_param( "inc_nodes_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "inc_beams_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "inc_shell_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "inc_solid_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "inc_th_sh_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "inc_extra_tg", 0, 0., "OFF", 4 )
jobfile.create_param( "Pri_During_Input", 0, 0., "ON", 4 )
jobfile.create_param( "Echo_File_Options", 0, 0., "Suppress Both", 4 )
jobfile.create_param( "Velocities", 0, 0., "OFF", 4 )
```

University of Pretoria etd – Kingsley, T C (2005)

```
jobfile.create_param( "Update_Beam", 0, 0., "OFF", 4 )
jobfile.create_param( "Interface_File", 0, 0., "", 3 )
jobfile.create_param( "Interface_File_Name", 0, 0., "INTFOR", 4 )
jobfile.create_param( "Time_Step", 0, 0., "ON", 4 )
jobfile.create_param( "Time_Interval", 0, 0., "", 1 )
jobfile.create_param( "Tolerance_Status", 0, 0., "OFF", 4 )
jobfile.create_param( "Tolerance_Value", 0, 0., "", 3 )
jobfile.create_param( "viewport", 0, 0., "default_viewport", 4 )
jobfile.create_param( "datbox_groupname", 0, 0., "default_group", 4 )
jobfile.create_param( "tranlational", 0, 0., "<0,0,0>", 4 )
jobfile.create_param( "rotational", 0, 0., "<0,0,0>", 4 )
jobfile.create_param( "loadcase_for_analysis", 0, 0., "Default", 4 )
jobfile.create_param( "set_node_opt", 0, 0., "YES", 4 )
jobfile.create_param( "set_beam_opt", 0, 0., "YES", 4 )
jobfile.create_param( "set_discrete_opt", 0, 0., "YES", 4 )
jobfile.create_param( "set_shell_opt", 0, 0., "YES", 4 )
jobfile.create_param( "set_solid_opt", 0, 0., "YES", 4 )
jobfile.create_param( "set_tshell_opt", 0, 0., "YES", 4 )
jobfile.create_param( "w_db_node", 5, 0., "", 1 )
jobfile.create_param( "w_db_beam", 5, 0., "", 1 )
jobfile.create_param( "w_db_discrete", 5, 0., "", 1 )
jobfile.create_param( "w_db_shell", 5, 0., "", 1 )
jobfile.create_param( "w_db_solid", 5, 0., "", 1 )
jobfile.create_param( "w_db_tshell", 5, 0., "", 1 )
jobfile.create_param( "all_groups_in_db_id_1", 0, 0., "baffles", 101 )
jobfile.create_param( "all_groups_in_db_id", 1, 0., "", 1 )
jobfile.create_param( "setcard_fullmodel_group_id_1", 0, 0., "baffles", 101 )
jobfile.create_param( "setcard_fullmodel_group_id", 1, 0., "", 1 )
jobfile.create_param( "factor_length_from_MADYMO", 0, 1., "", 3 )
jobfile.create_param( "factor_on_time_from_MADYMO", 0, 1., "", 3 )
jobfile.create_param( "factor_on_force_from_MADYMO", 0, 1., "", 3 )
jobfile.create_param( "Wait_time_as_MADYMO_computes", 0, 0., "", 3 )
jobfile.create_param( "Flip_X_coord_of_MADYMO", 0, 0., "OFF", 4 )
jobfile.create_param( "Flip_Y_coord_of_MADYMO", 0, 0., "OFF", 4 )
jobfile.create_param( "Flip_Z_coord_of_MADYMO", 0, 0., "OFF", 4 )
jobfile.create_param( "Num_Dyna_steps_per_MADYMO_step", 1, 0., "", 1 )
jobfile.create_param( "filename_interface", 0, 0.,  @
"Default_interfacefile.isf1", 4 )
jobfile.create_param( "restart_label_file", 0, 0., "", 4 )
jobfile.create_param( "DTIwrite_switch_id", 0, 0., "End Deck", 4 )
jobfile.create_param( "DTIdirect_text_toggle_id", 0, 0., "OFF", 4 )
jobfile.close(  )
uil_file_close.go(  )
sys_library( "add", "lsdyna3d.plb" )
lsdyna3d_spawn_generic( "pat3lsdyna", " -d patran_tester1.db -j patran_keyword", TRUE
)
$uil_file_open.go( "patran_tester1.db" )
$
$
$                                    END
$
```

APPENDIX T: Mesh file cleaning SED file

```
1,28 d
/SECTION_SHELL$/{
n
n
d
}
/SECTION_SHELL$/{
n
d
}
/SECTION_SHELL$/{
d
}
/*PART$/{
n
n
d
}
/*PART$/{
n
d
}
/*PART$/{
d
}
/SECTION_SOLID$/{
n
d
}
/SECTION_SOLID$/{
d
}
/Material :/,$ d
w mesh.k
```

APPENDIX U: LS-DYNA keyword file (Model settings section only)

```
*KEYWORD 100000000
$ continuum control
*CONTROL_ALE
2,1,2,-1.00,.000E+00,.000E+00,.000E+00,.000E+00
.000E+00,.000E+00,.000E+00,.000E+00,.000E+00,0
*CONTROL_CPU
18e3
*CONTROL_TERMINATION
<<term_time>>,0,.000E+00,.000E+00,.000E+00
*DATABASE_BINARY_D3PLOT
<<d3_interval>>,0
*DATABASE_BINARY_D3DUMP
<<dump_interval>>
$ define degree of freedom
*BOUNDARY_PRESCRIBED_MOTION_NODE
1,1,1,3,1.00,0,0.000,0.000
*load_body_z
2,9.81
*set_part_list
2
3,4
*set_part_list
1
1,2
*DATABASE_HISTORY_SHELL
$id1,id2,id3,id4,id5,id6,id7,id8
$
20000,20001,20002,20003
$
*DATABASE_HISTORY_NODE
1
$3318,4578,5838,7098,3977,5237,6497,7757
*DATABASE_ELOUT
$       dt
    0.0001         3
*DATABASE_NODOUT
    0.0001         3
*CONSTRAINED_LAGRANGE_IN_SOLID
$ Coupling control (penalty/energy)
         2         2         1         0         2         4         2
1
$    START       END
                                               0.3         0
                                       1
*ALE_MULTI-MATERIAL_GROUP
$ group mixable materials (air/water)
         3         1
         4         1
*ALE_REFERENCE_SYSTEM_GROUP
$ set group that will follow local coordinate system
2,0,5,1
0
*ALE_REFERENCE_SYSTEM_NODE
$ define local coordinate system RHR (x1,x2,x3)
1
5000,1,5100
0
$
$  Rigid
$
*MAT_RIGID
1,7.830E+03,2.070E+11,0.300,0.
```

```
$ constraint directions (translation y-z)(all rotation)
1.0,5.0,7.0
0
*HOURGLASS
$ Bulk viscosity properties
1,0,0.,0,0.,0.
*SECTION_SHELL
$ 1 integration point (no bending)
1,2,0.,1.,0.,0.,0
1.000E-02,1.000E-002,1.000E-02,1.000E-02,0.
*PART
material type # 20 (Rigid)
1,1,1,0,1,0
$
$  Baffles
$
*MAT_PLASTIC_KINEMATIC
2,7.830E+03,2.070E+11,0.300,5.000E+09,100.,0.
0.,0.,0.
*HOURGLASS
2,0,0.,0,0.,0.
*SECTION_SHELL
2,2,0.,0.,0.,0.,0
$ Baffle thickness
<<baffle_T>>,<<baffle_T>>,<<baffle_T>>,<<baffle_T>>,0.
*PART
material type # 3 (Kinematic/Isotropic Elastic-Plastic)
2,2,2,0,2,0
$
$ Air
$
*MAT_NULL
         4    1.1845       0.0 0.0000184       0.0       0.0       0.0
0.0
*HOURGLASS
4,0,0.,0,0.,0.
*EOS_LINEAR_POLYNOMIAL_WITH_ENERGY_LEAK
4,0.0,0.0,0.0,0.0,0.4,0.4,0.0
,1.0,0
*SECTION_SOLID
4,11,0
*PART
outer air (kg-m-s)
3,4,4,4,4,0
$
$ Water
$
*MAT_NULL
         3     998.0       0.0     0.001       0.0       0.0       0.0
0.0
*HOURGLASS
3,0,0.,0,0.,0.
*EOS_GRUNEISEN
3,1647.0,1.921,-0.096,0.0,0.35,0.0,0.0
0.0
*SECTION_SOLID
3,11,0
*PART
water (kg-m-s)
4,3,3,3,3,0
$
*INCLUDE
mesh.k
*END

$
$ LOAD CURVES
$
```

```
*DEFINE_CURVE
1,0,.000E+00,.000E+00,.000E+00,.000E+00,0
.0000000000000E+00,800.0000000000
1.0000000474975E-03,.0000000000000E+00
5.000000000000,.0000000000000E+00
*DEFINE_CURVE
2,0,.000E+00,.000E+00,.000E+00,.000E+00,0
.0000000000000E+00,1.000000000000
5.000000000000,1.000000000000
*DEFINE_CURVE
3,0,.000E+00,.000E+00,.000E+00,.000E+00,0
0,20.34915942
0.000005,22.56476372
…
… (acceleration data)
…
0.01,0
0.09,0
*END
```

APPENDIX V: LSOPT command file – 3D impact only optimisation

```
"Structural optimisation of 3D baffled tank to reduce mass and maintain structural
integrity."
Author "K Craig, T C Kingsley"
$ Created on Sat Nov  6 11:31:41 2004
solvers 1
responses 4
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 3
 Variable 'baffle_H' 0.1
  Lower bound variable 'baffle_H' 0.08
  Upper bound variable 'baffle_H' 0.3
 Variable 'hole_D' 0.025
  Lower bound variable 'hole_D' 0.015
  Upper bound variable 'hole_D' 0.05
 Variable 'baffle_T' 0.002
  Lower bound variable 'baffle_T' 0.001
  Upper bound variable 'baffle_T' 0.01
$
$ CONSTANTS
$
constants 3
 Constant 'term_time' 0.015
 Constant 'd3_interval' 0.002
 Constant 'dump_interval' 0.005

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "Patran-Dyna"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "Patran-Dyna"
$
 solver dyna960 'Patran-Dyna'
  solver command "ls970"
  solver input file "header.k"
  solver append file "footer.k"
  prepro own
  prepro command "../../process"
  prepro input file "prepro_pre_tank.ses"
  solver order linear
  solver experiment design dopt
  solver number experiments 7
  solver basis experiment 3toK
  solver concurrent jobs 1
$
$ RESPONSES FOR SOLVER "Patran-Dyna"
$
 response 'Max_PriStress_baff' 1 0 "DynaPStress S1 2 MAX"
 response 'baffle_mass' 1 0 "DynaMass 2 MASS"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "Patran-Dyna"
$
 response 'too_big_hole' expression {(baffle_H/2)-hole_D}
 response 'baffle_mass_calc' expression {7830*baffle_T*(4*(baffle_H*0.4-
(2*(3.1415926536*hole_D**2))))}

$
$ OBJECTIVE FUNCTIONS
```

```
$
 objectives 1
 objective 'baffle_mass_calc' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 2
 constraint 'Max_PriStress_baff'
  strict
  upper bound constraint 'Max_PriStress_baff' 2e+08
  move
 constraint 'too_big_hole'
  lower bound constraint 'too_big_hole' 0.03
  slack
$
$ JOB INFO
$
 iterate param design 0.01
 iterate param objective 0.01
 iterate param stoppingtype and
 iterate 6
STOP
```

University of Pretoria etd – Kingsley, T C (2005)

APPENDIX W: LSOPT command file – 2D extruded impact only optimisation

```
"Impact analysis of baffled liquid container (Effective element stress)"
Author "Thomas Kingsley, Ken Craig"
$ Created on Mon Jul 26 08:17:05 2004
solvers 1
responses 10
$
$ NO HISTORIES ARE DEFINED
$
$
$ DESIGN VARIABLES
$
variables 3
 Variable 'mid_baf_centroid' 100
  Lower bound variable 'mid_baf_centroid' 60
  Upper bound variable 'mid_baf_centroid' 320
 Variable 'mid_baf_height' 100
  Lower bound variable 'mid_baf_height' 40
  Upper bound variable 'mid_baf_height' 320
 Variable 'baffle_T' 0.008
  Lower bound variable 'baffle_T' 0.001
  Upper bound variable 'baffle_T' 0.015
$
$ CONSTANTS
$
constants 3
 Constant 'term_time' 0.015
 Constant 'd3_interval' 0.001
 Constant 'dump_interval' 0.005
$
$ DEPENDENT VARIABLES
$
dependent 2
 Dependent 'baffle_H' {mid_baf_height/1000}
 Dependent 'baffle_centroid' {mid_baf_centroid/1000}

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "Patran-Dyna"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "Patran-Dyna"
$
 solver dyna960 'Patran-Dyna'
  solver command "../../dyna_script"
  solver input file "header.k"
  solver append file "footer.k"
  prepro own
  prepro command "../../process"
  prepro input file "pat_prepro_2Dtank.ses"
  solver order linear
  solver experiment design dopt
  solver number experiments 7
  solver basis experiment 3toK
  solver concurrent jobs 2
$
$ RESPONSES FOR SOLVER "Patran-Dyna"
$
 response 'max_eff_stress_20000b' 1 0 "DynaASCII Elout E_STRESS 20000 2 MAX 0 0.1"
 response 'max_eff_stress_20000a' 1 0 "DynaASCII Elout E_STRESS 20000 1 MAX 0 0.1"
 response 'max_eff_stress_20001a' 1 0 "DynaASCII Elout E_STRESS 20001 1 MAX 0 0.1"
 response 'max_eff_stress_20001b' 1 0 "DynaASCII Elout E_STRESS 20001 2 MAX 0 0.1"
 response 'max_eff_stress_20002b' 1 0 "DynaASCII Elout E_STRESS 20002 2 MAX 0 0.1"
 response 'max_eff_stress_20002a' 1 0 "DynaASCII Elout E_STRESS 20002 1 MAX 0 0.1"
 response 'max_eff_stress_20003b' 1 0 "DynaASCII Elout E_STRESS 20003 2 MAX 0 0.1"
```

University of Pretoria etd – Kingsley, T C (2005)

```
 response 'max_eff_stress_20003a' 1 0 "DynaASCII Elout E_STRESS 20003 1 MAX 0 0.1"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "Patran-Dyna"
$
 response 'baffle_upper' expression {(mid_baf_height/2)+mid_baf_centroid}
 response 'baffle_lower' expression {mid_baf_centroid-(mid_baf_height/2)}

composites 9
$
$ COMPOSITE RESPONSES
$
 composite 'Max_eff_stress_scaled_0a' type weighted
  composite 'Max_eff_stress_scaled_0a' response 'max_eff_stress_20000a' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_0b' type weighted
  composite 'Max_eff_stress_scaled_0b' response 'max_eff_stress_20000b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_1b' type weighted
  composite 'Max_eff_stress_scaled_1b' response 'max_eff_stress_20001b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_1a' type weighted
  composite 'Max_eff_stress_scaled_1a' response 'max_eff_stress_20001a' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_2a' type weighted
  composite 'Max_eff_stress_scaled_2a' response 'max_eff_stress_20002a' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_2b' type weighted
  composite 'Max_eff_stress_scaled_2b' response 'max_eff_stress_20002b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_3b' type weighted
  composite 'Max_eff_stress_scaled_3b' response 'max_eff_stress_20003b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_3a' type weighted
  composite 'Max_eff_stress_scaled_3a' response 'max_eff_stress_20003a' 1 scale 1e+07
$
$ COMPOSITE EXPRESSIONS
$
 composite 'Baffle_Mass' {10*(7830*baffle_T*0.4*((mid_baf_height/1000)+2*0.01))}
$
$ OBJECTIVE FUNCTIONS
$
 objectives 1
 objective 'Baffle_Mass' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 10
  move
 constraint 'baffle_upper'
  strict
  upper bound constraint 'baffle_upper' 340
 constraint 'baffle_lower'
  lower bound constraint 'baffle_lower' 20
  slack
  stay
 constraint 'Max_eff_stress_scaled_0a'
  strict
  upper bound constraint 'Max_eff_stress_scaled_0a' 20
 constraint 'Max_eff_stress_scaled_0b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_0b' 20
 constraint 'Max_eff_stress_scaled_1b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_1b' 20
 constraint 'Max_eff_stress_scaled_1a'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_1a' 20
 constraint 'Max_eff_stress_scaled_2a'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_2a' 20
 constraint 'Max_eff_stress_scaled_2b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_2b' 20
 constraint 'Max_eff_stress_scaled_3b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_3b' 20
 constraint 'Max_eff_stress_scaled_3a'
  slack
```

University of Pretoria etd – Kingsley, T C (2005)

```
  strict
  upper bound constraint 'Max_eff_stress_scaled_3a' 20
$
$ JOB INFO
$
 concurrent jobs 1
 iterate param design 0.01
 iterate param objective 0.01
 iterate param stoppingtype and
 iterate 8
STOP
```

APPENDIX X: LSOPT command file – Multidisciplinary optimisation (Sloshing and impact)

```
"MDO of baffled liquid container (scaled TDV and effective element stress)"
Author "Thomas Kingsley, Ken Craig"
$ Created on Mon Sep 13 10:16:44 2004
solvers 2
responses 16
$
$ NO HISTORIES ARE DEFINED
$
$ DESIGN VARIABLES
$
variables 5
 Variable 'mid_baf_centroid' 100
  Lower bound variable 'mid_baf_centroid' 60
  Upper bound variable 'mid_baf_centroid' 320
 Variable 'side_baf_centroid' 100
  Lower bound variable 'side_baf_centroid' 15
  Upper bound variable 'side_baf_centroid' 185
  Local 'side_baf_centroid'
 Variable 'mid_baf_height' 100
  Lower bound variable 'mid_baf_height' 40
  Upper bound variable 'mid_baf_height' 320
 Variable 'side_baf_width' 100
  Lower bound variable 'side_baf_width' 10
  Upper bound variable 'side_baf_width' 180
  Local 'side_baf_width'
 Variable 'baffle_T' 0.008
  Lower bound variable 'baffle_T' 0.001
  Upper bound variable 'baffle_T' 0.015
  Local 'baffle_T'
$
$ CONSTANTS
$
constants 4
 Constant 'fill_level' 0.28
 Constant 'term_time' 0.015
 Constant 'd3_interval' 0.001
 Constant 'dump_interval' 0.005
$
$ DEPENDENT VARIABLES
$
dependent 4
 Dependent 'baffle_H' {mid_baf_height/1000}
 Dependent 'baffle_centroid' {mid_baf_centroid/1000}
 Dependent 'side_baffle_W' {side_baf_width/1000}
 Dependent 'side_baffle_cent' {side_baf_centroid/1000}

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      OPTIMIZATION METHOD
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
Optimization Method SRSM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "fluent"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "fluent"
$
 solver own 'fluent'
  solver command "../../fluent_script"
  solver input file "fluentjou2D.jou"
  prepro own
  prepro command "gambit -inp"
  prepro input file "2D_tom_gambit.jou"
  solver order linear
  solver experiment design dopt
  solver basis experiment 3toK
  solver concurrent jobs 1
$
```

```
$ LOCAL DESIGN VARIABLES FOR SOLVER "fluent"
$
  solver variable 'side_baf_centroid'
  solver variable 'side_baf_width'
$
$ RESPONSES FOR SOLVER "fluent"
$
 response 'TDV' 1000 0 "cat TotalDev.txt"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "fluent"
$
 response 'SB_side_constraint' expression {side_baf_centroid-side_baf_width/2}
 response 'SB_mid_constaint' expression {side_baf_centroid+side_baf_width/2}
 response 'MB_up_constaint' expression {mid_baf_centroid+mid_baf_height/2}
 response 'MB_low_constraint' expression {mid_baf_centroid-mid_baf_height/2}
 response 'baffle_length' expression {mid_baf_height+2*side_baf_width}

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      SOLVER "Patran-Dyna"
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ DEFINITION OF SOLVER "Patran-Dyna"
$
 solver dyna960 'Patran-Dyna'
  solver command "../../dyna_script"
  solver input file "header.k"
  solver append file "footer.k"
  prepro own
  prepro command "../../process"
  prepro input file "pat_prepro_2Dtank.ses"
  solver order linear
  solver experiment design dopt
  solver basis experiment 3toK
  solver concurrent jobs 2
$
$ LOCAL DESIGN VARIABLES FOR SOLVER "Patran-Dyna"
$
  solver variable 'baffle_T'
$
$ RESPONSES FOR SOLVER "Patran-Dyna"
$
 response 'max_eff_stress_20000b' 1 0 "DynaASCII Elout E_STRESS 20000 2 MAX 0 0.1"
 response 'max_eff_stress_20000a' 1 0 "DynaASCII Elout E_STRESS 20000 1 MAX 0 0.1"
 response 'max_eff_stress_20001a' 1 0 "DynaASCII Elout E_STRESS 20001 1 MAX 0 0.1"
 response 'max_eff_stress_20001b' 1 0 "DynaASCII Elout E_STRESS 20001 2 MAX 0 0.1"
 response 'max_eff_stress_20002b' 1 0 "DynaASCII Elout E_STRESS 20002 2 MAX 0 0.1"
 response 'max_eff_stress_20002a' 1 0 "DynaASCII Elout E_STRESS 20002 1 MAX 0 0.1"
 response 'max_eff_stress_20003b' 1 0 "DynaASCII Elout E_STRESS 20003 2 MAX 0 0.1"
 response 'max_eff_stress_20003a' 1 0 "DynaASCII Elout E_STRESS 20003 1 MAX 0 0.1"
$
$ RESPONSE EXPRESSIONS FOR SOLVER "Patran-Dyna"
$
 response 'baffle_upper' expression {(mid_baf_height/2)+mid_baf_centroid}
 response 'baffle_lower' expression {mid_baf_centroid-(mid_baf_height/2)}

composites 9
$
$ COMPOSITE RESPONSES
$
 composite 'Max_eff_stress_scaled_0a' type weighted
  composite 'Max_eff_stress_scaled_0a' response 'max_eff_stress_20000a' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_0b' type weighted
  composite 'Max_eff_stress_scaled_0b' response 'max_eff_stress_20000b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_1b' type weighted
  composite 'Max_eff_stress_scaled_1b' response 'max_eff_stress_20001b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_1a' type weighted
  composite 'Max_eff_stress_scaled_1a' response 'max_eff_stress_20001a' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_2a' type weighted
  composite 'Max_eff_stress_scaled_2a' response 'max_eff_stress_20002a' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_2b' type weighted
  composite 'Max_eff_stress_scaled_2b' response 'max_eff_stress_20002b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_3b' type weighted
  composite 'Max_eff_stress_scaled_3b' response 'max_eff_stress_20003b' 1 scale 1e+07
 composite 'Max_eff_stress_scaled_3a' type weighted
  composite 'Max_eff_stress_scaled_3a' response 'max_eff_stress_20003a' 1 scale 1e+07
$
$ COMPOSITE EXPRESSIONS
```

University of Pretoria etd – Kingsley, T C (2005)

```
$
 composite 'Baffle_Mass' {10*(7830*baffle_T*0.4*(baffle_H+2*side_baffle_W))}
$
$ OBJECTIVE FUNCTIONS
$
 objectives 2
 objective 'TDV' 1
 objective 'Baffle_Mass' 1
$
$ CONSTRAINT DEFINITIONS
$
 constraints 15
  move
 constraint 'SB_side_constraint'
  strict
  lower bound constraint 'SB_side_constraint' 10
  slack
 constraint 'SB_mid_constaint'
  strict
  upper bound constraint 'SB_mid_constaint' 190
 constraint 'MB_up_constaint'
  slack
  strict
  upper bound constraint 'MB_up_constaint' 340
 constraint 'MB_low_constraint'
  lower bound constraint 'MB_low_constraint' 40
  slack
  stay
 constraint 'baffle_length'
  lower bound constraint 'baffle_length' 0
  upper bound constraint 'baffle_length' 540
  move
 constraint 'baffle_upper'
  strict
  upper bound constraint 'baffle_upper' 340
 constraint 'baffle_lower'
  lower bound constraint 'baffle_lower' 20
  slack
  stay
 constraint 'Max_eff_stress_scaled_0a'
  strict
  upper bound constraint 'Max_eff_stress_scaled_0a' 20
 constraint 'Max_eff_stress_scaled_0b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_0b' 20
 constraint 'Max_eff_stress_scaled_1b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_1b' 20
 constraint 'Max_eff_stress_scaled_1a'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_1a' 20
 constraint 'Max_eff_stress_scaled_2a'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_2a' 20
 constraint 'Max_eff_stress_scaled_2b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_2b' 20
 constraint 'Max_eff_stress_scaled_3b'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_3b' 20
 constraint 'Max_eff_stress_scaled_3a'
  slack
  strict
  upper bound constraint 'Max_eff_stress_scaled_3a' 20
$
$ JOB INFO
iterate param design 0.01
 iterate param objective 0.01
 iterate param stoppingtype and
 iterate 15
STOP
```