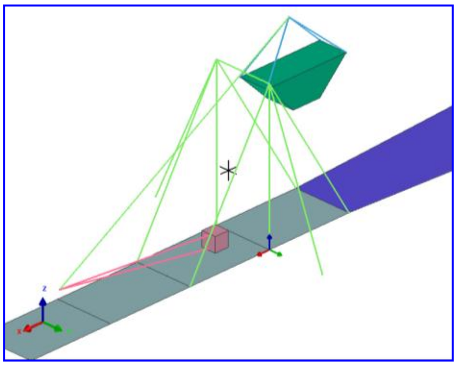
Tutorial 14

Optimisation of a medieval catapult (Trebuchet) system to hit a target



Problem description

Outline	A medieval catapult system (Trebuchet) is modelled and used as a basis for optimisation using PAM-OPT and PAM-CRASH. Parameters such as counterweight and release point are optimised to achieve a desired height and throwing distance. The model is used to demonstrate the structure of a PAM-OPT analysis. Different optimisation methods are compared including Gradient, RSM and Genetic methods. For general information on a catapult (Trebuchet) see http://en.wikipedia.org/wiki/Trebuchet
Analysis type(s):	FE explicit using PAM-CRASH coupled to PAM-OPT and Visual-Viewer used in batch mode
Element type(s):	Bars, shells and solids
Materials law(s):	Elastic and elasto-plastic for wall impact
Model options:	Boundary conditions, gravity loading, coupled sensors activation and deactivation, rigid bodies, contact interfaces
Key results:	Optimisation of height and distance to a target point
Prepared by:	Anthony Pickett, ESI GmbH/Institute for Aircraft Design, Stuttgart
Date:	August 2011

Background information

Pre-processor, Solver, Post-processor and Optimisation codes used:

- **Visual-Crash:** To assign part and material data, rigid body constraints, contacts, sensors, gravity, time history and analysis control data.
- **Analysis (PAM-CRASH Explicit):** To perform the explicit Finite Element analysis.
- **Visual-Viewer (in batch mode):** To evaluate results of the projectile trajectory (xxx.THP file).
- **PAM-OPT:** To optimise throwing distance to a specified target point.

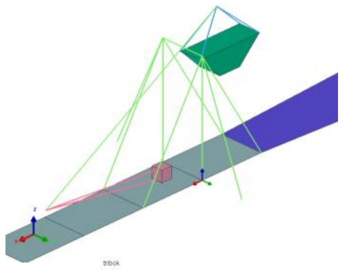
Prior knowledge for the exercise:

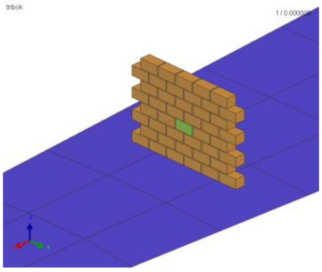
No prior PAM-OPT knowledge is required for working through this exercise; but it is assumed the user is reasonably familiar with Visual-Crash, Visual-Viewer and PAM-CRASH.

Problem data and description

Units: kN, mm, kg, msec

Description: The catapult is constructed from bar and beam elements having approximate wood properties.
The sling is modelled with stiff bar elements.
The counterweight uses shell elements with an added mass of typically 18000kg.
The floor is modelled as 2D shell elements and the impact wall uses 3D solid elements.





Loading: Gravity loading forces the counterweight downward and starts the throwing stone (projectile) into motion.

Material: Elastic materials or rigid bodies are used throughout, except for a soft elasto-plastic law for the wall bricks to give a non-elastic (energy absorbing) contact.

Tutorial 14: General exercise in optimisation using PAM-OPT

Supplied datasets

All datasets are provided for this exercise and there is no need to create any new Finite Element meshes.

The intention is that necessary PAM-OPT files are constructed from a PAM-CRASH model provided and that the Visual-Viewer command files (so called template files) are created from scratch. However, if things really go wrong, all datasets are supplied as detailed in the Table below.

Directory	**Files**	**Purpose**
Catapult_OriginalModel	1. Catapult_Original	• The original model
Catapult_ModifiedForOptimisation	1. Catapult_ModifiedForOptimisation.pc 2. Catapult_ModifiedForOptimisation_WoodModifiedTimestep.pc	• The simplified model, after modifications, suitable for optimisation studies • Further improvements to lower CPU costs for optimisation studies
Catapult_OptimiseModel1_Gradient	1. Catapult_Opt1.CDS 2. Visual.tpl	• The PAM-OPT file for Gradient optimisation • The visual template file for Visual-Viewer batch mode post-processing
Catapult_OptimiseModel1_RMS	1. Catapult_Opt1.CDS 2. Visual.tpl	• The PAM-OPT file for RMS optimisation • The visual template file for Visual-Viewer batch mode post-processing
Catapult_OptimiseModel1_Genetic	1. Catapult_Opt1.CDS 2. Visual.tpl	• The PAM-OPT file for Genetic optimisation • The visual template file for Visual-Viewer batch mode post-processing
Various directories	Various xxx.CDS and xxx.tpl files	For parts 5,6 and 7 of this tutorial exercise

Part 1: Setting up the analysis model

Preparing the FE analysis model for optimisation

Most Finite Element models are not directly suitable for optimisation! The optimisation process may require many (perhaps hundreds) of simulations to be performed and therefore an important aspect of preparation work is to try and simplify the model for two reasons:

1. **CPU efficiency:** Ideally the models should run in seconds or less than a few minutes; but clearly this depends on the CPU power available and the CPU time you are prepared to accept.
2. **Robustness:** Try to simplify the model and have a reliable stable simulation. At this stage you should have an idea which features in the model will be optimised (varied); and make sure the model works well (stable and CPU efficient) over the range that these parameters will be varied.

As an example it is worth studying the original and simplified models of the catapult that was modified here to make it more suitable for optimisation. Note that the original model was already reasonably simplified and could have been much more complex!

1.1 Original model (Catapult_Original.pc)

Start Visual-PAM and open this dataset; then study the modelling methods used. In particular look at:

Parts	Thirteen parts are defined and linked to materials and other options.
Materials	Eight materials are defined for the full catapult model.
Applied loading	This is in the form of gravity and added mass to the counterweight box: Gravity: See **Loads > Structural loads > Acceleration Field** Added mass: See **Loads > Structural Loads > Non Structural Mass.**
Boundary conditions	The floor and base of the catapult are fully fixed: See **Loads > Structural loads > Displacement BC.**
Contacts	Three contacts are defined: 1. A non-symmetric contact for the projectile to the ground (type 34) 2. A self contact for the projectile, the wall bricks and the floor (type 36) 3. A contact between the projectile and the wall facing; this is used in a contact trigger sensor to deactivate the ties holding the bricks together (type 34).
Rigid bodies	See **Constraints > Rigid bodies**. An important trick here is that the stone projectile is always treated as a Rigid body. First it is a rigid body (Nr. 2) connected to the sling connections; this is deactivated and released by a sensor. At the same time the stone alone is reactivated as a new rigid body (Nr.3); again by the sensor. This rigid body treatment avoids bad element distortions that would occur if it was treated as a normal element having large rotations/displacements.
Outputs	One node on the projectile and the release hook element are specified under **Outputs** for extra information stored in the .THP file.
Sensors	See **Auxiliaries > Sensor**. A first sensor is used to release the projectile and is controlled by extension of an element at the end of the catapult arm (Sensor 1). A second contact force activated sensor (Nr. 2) is activated once the projectile hits the wall (this is linked to contact definition Nr 3).
Rupture	A rupture model is defined to hold the wall bricks together and approximate the cement.

CPU performance: 1779 seconds with a stable timestep of 50 μsec

1.2 Modified model (Catapult_ModifiedForOptimisation.pc): Changes made for simplification and robustness

Read this dataset into Visual-PAM. You will find the following changes have been made to improve stability and CPU performance:

- The sling system comprising of several bars and a release beam are changed to a simple sling comprising of only 2 long bars.
- Gravity was only applied to the counterweight stone, the catapult and the stone projectile.
- The original wall was replaced with a simpler wall of larger stones having a greater timestep.
- The density of the catapult wood was raised to increase the stable timestep. Also, the stone impact wall has increased density and lower stiffness to improve the timestep.
- Various other changes were made to the contacts and sensors.
- Release of the stone is controlled by a sensor (relative distance) between a node on the projectile and a fixed node on the ground.

CPU performance: These changes increased the timestep to 208 μsec and reduced the CPU to 54 sec.

Note that these modifications allow the model to run 33 times faster and gives essentially the same throwing distance.

1.3 Further model changes (Catapult_ModifiedForOptimisation_WoodModified.pc) for timestep/CPU improvements

For the purpose of this tutorial the CPU is further reduced by changing the catapult (wood) mechanical properties as follows:

- Density 0.8E-06 kg/mm^3 increased to 5.0E-06 kg/mm^3
- Modulus 15 kN/mm^2 reduced to 5 kN/mm^2

CPU performance: These changes increased the timestep to 900 μsec and reduced the CPU to 16 sec.

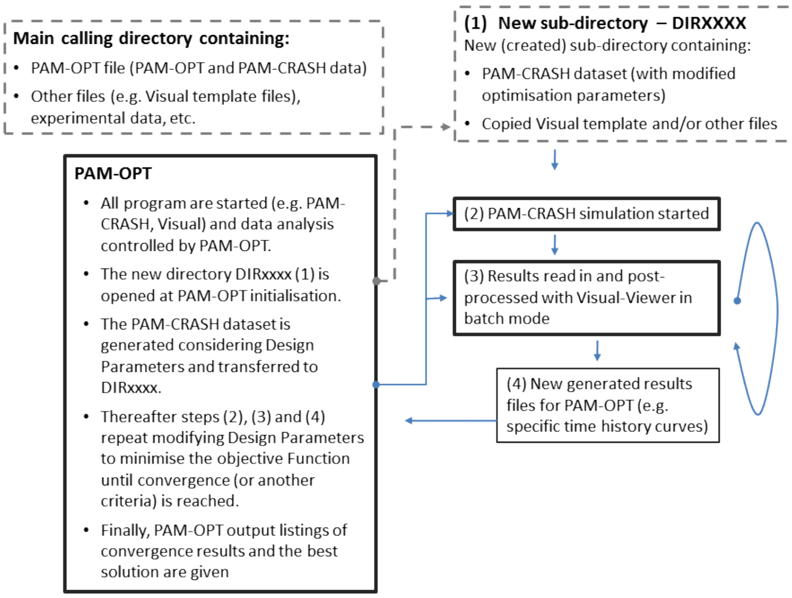
Note that the heavier, more flexible wood does, as would be expected, significantly change (reduce) the throwing distance compared to the previous models.

Part 2: Preparing the files for PAM-OPT

2.1 Some general comments on the PAM-OPT file structure and optimisation

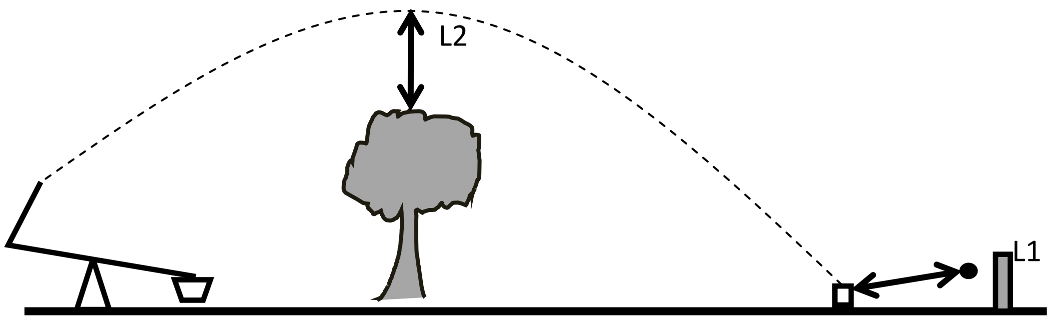
The file structure of a typical PAM-OPT simulation using PAM-CRASH for the analysis and Visual-Viewer for the post-processing is shown below. The main PAM-OPT dataset is held in a main directory and comprises of two parts; first a section to control the PAM-OPT operations, and second a part which is essentially the standard PAM-CRASH dataset. This dataset is read by PAM-OPT. Also, held in this directory are the template (or other) files used to control Visual Viewer.

Upon starting PAM-OPT a new sub-directory is automatically created (DIRXXX) and the modified PAM-CRASH, visual and possible other files are copied to this directory by PAM-OPT. The optimisation procedure then performs successive analyses. In each case specific results are generated by Visual-Viewer which is then read in by PAM-OPT. The results may be further analysed in PAM-OPT and finally used in an Objective Function expression to perform a minimisation and thereby try to determine 'improved' models for further analysis and evaluation.



2.2 The objective function

The optimisation is formulated with an Objective Function that has to be minimised. It may contain only one parameter for minimisation (single objective minimisation; e.g. minimisation of distance L1 to the target), or could contain several parameters (multi-objective minimisation; e.g. minimisation of L1 and L2 so a certain height is also reached). In this exercise design parameters that could be varied, to minimise the Objective Function, are weight of the counterweight and/or release point of the stone from the sling.



2.3 Preparing the PAM-CRASH model for this optimisation exercise

The safest start is as follows:

1. Create a new directory method to (e.g. **Catapult_OptimiseModel1_Gradient**)
2. Copy the previous dataset (Catapult_ModifiedForOptimisation_WoodModified.pc) to the new directory and give it a simpler name (e.g. Catapult_Opt1.pc).
3. Make the following two changes to Catapult_Opt1.pc:
   i. First, reposition the wall for less 'throwing' distance to help further reduce CPU time; edit **Extra Controls >Transform > wall pos** and change the TRANS_d value from -270000 to -170000. The wall should then be positioned closer to the centre of the ground area.
   ii. Second, increase the run time by changing **Controls > Advanced controls > RUNEND** from 10000 to 15000. This will help ensure the projectile always passes the wall location.
4. Save and run this model to get the Catapult_Opt1.DSY and Catapult_Opt1.THP results files.
5. Check it runs successfully to the end (look for 'NORMAL TERMINATION' in the Catapult_Opt1.out file). The Catapult_Opt1.DSY file can be loaded into Visual-Viewer and should show the stone bounces a few times and missing the wall. Clearly some optimisation is needed to get it to hit the target!

2.4 Preparing the Visual-Viewer batch control (xxx.tpl) file

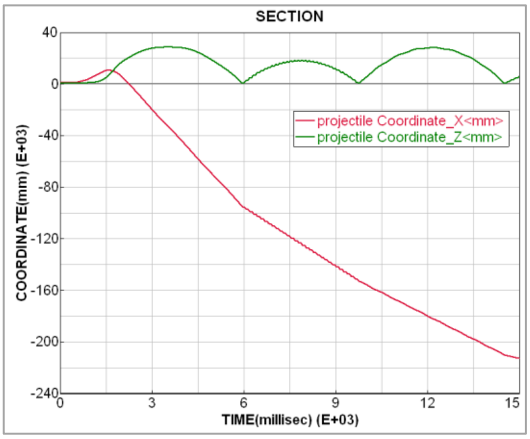
PAM-OPT will run both PAM-CRASH and Visual-Viewer in batch mode. The series of commands to control Visual-Viewer are stored in a command (template xxx.tpl) file that is repeatedly used to read the xxx.THP file, generate curves and write out required information.

For this study we are interested to get Visual-Viewer to generate and store a time history file of distance of the projectile to a target point on the wall. Later this information will be read by PAM-OPT to find the minimum distance.

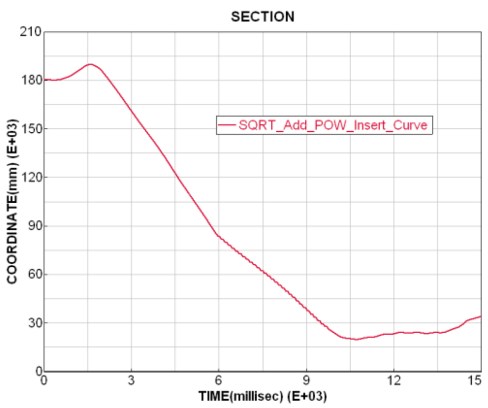
For generation and output of the 'minimum distance to the target point' time history file we proceed as follows:

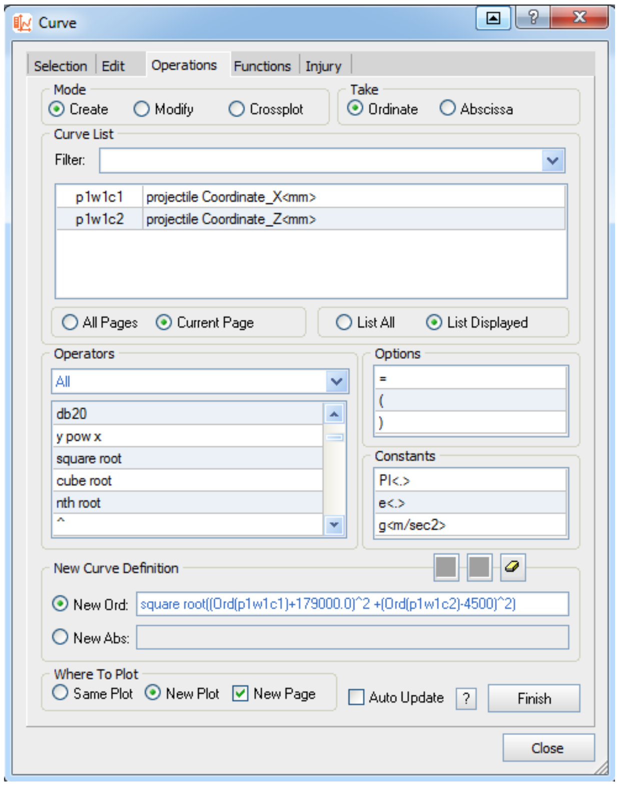
1. First identify the desired target point. Check using Visual-PAM with the Catapult_Opt1.pc file that sensible coordinates for a central impact point are approximately:
   X = -179000
   Y = 0
   Z = +4500

2. Start Visual-Viewer and read in the Catapult_Opt1.THP
3. Plot a time history of the projectile node (coordinate x) and also a time history of the projectile node (coordinate z). These curves will be used to construct the required distance to target point time history curve.



4. Activate **Curves > Arithmetic Function** and then the tab **Operations**. The two (x,z) coordinate curves appear in the new panel. A new function is then defined { square root((Ord(p1w1c1)+179000.0)^2 +(Ord(p1w1c2)-4500)^2) } in the **New Coordinate** definition using these curves and the target wall location. Also, activate **New Plot** and **New Page**. Finally, **Finish** to get the new time history plot of distance of projectile from the target point.

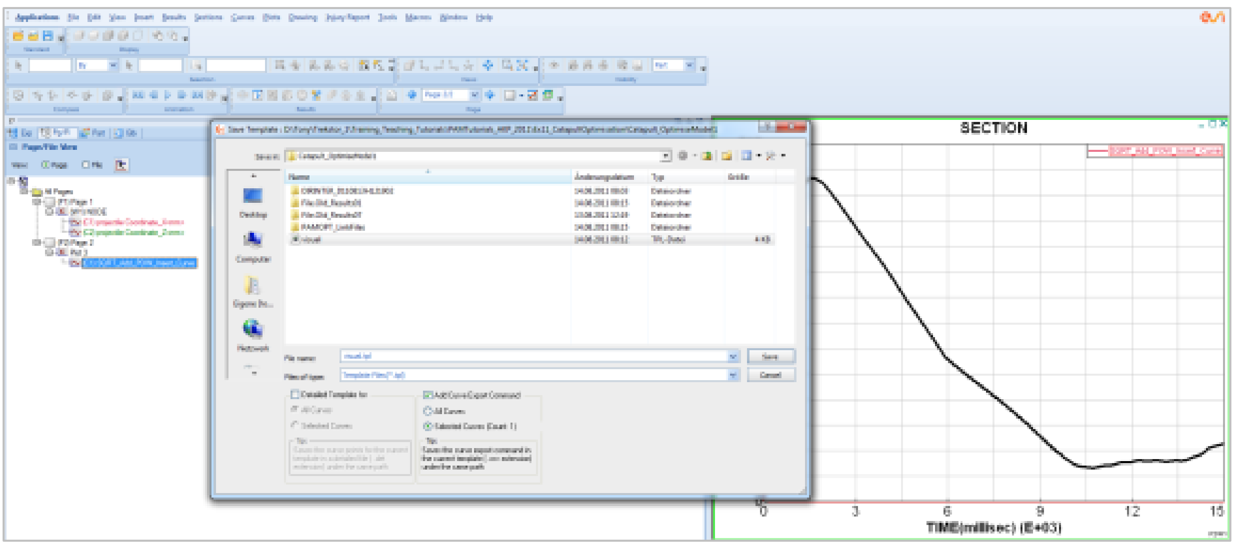




5. Now the procedure so far used to get get this time history plot with Visual-Viewer has to be stored. First make sure the curve you want to save in the template file is highlighted in black by simply clicking on it (it will appear black). Then activate **File > Save Template** to get the panel shown below.

   Loacate your working directory at the top and activate the '**Add Curve Export Command**' and **Selected Curves**. Note this will save the new curve in the directory; but ONLY when this xxx.tpl file is activated. Check that the selected curves count = 1.

   Finally, give a file name for the template file (e.g. visual) and close the action with **Save**.



6. The Visual-Viewer session is now finished. Close as usual with **File > Exit**.
7. One final operation is needed: The top and bottom of the new visual.tpl file has to be edited to have the input and output file locations modified as shown below. Use any editor, make the changes indicated below, save and close.

Old (top) visiual.tpl file information
+GLOBAL_VARIABLES_START
FILE0 =
"D:/Tony/Trekstor_1/Training_Teaching_Tutorials/PAMTutorials_AKP_2011/Ex11_CatapultOptimisatio
n/Catapult_OptimiseModel1_Gradient/Catapult_Opt1.THP"
+GLOBAL_VARIABLES_END

New (top) visiual.tpl file information
+GLOBAL_VARIABLES_START
FILE0 = "Catapult_Opt1.CDS.THP"
+GLOBAL_VARIABLES_END

Old (bottom) visiual.tpl file information
+ScriptStart
+curvexportcrv('D:/Tony/Trekstor_1/Training_Teaching_Tutorials/PAMTutorials_AKP_2011/Ex11_Cata
pultOptimisation/Catapult_OptimiseModel1_Gradient/visual.crv', 'ALL')

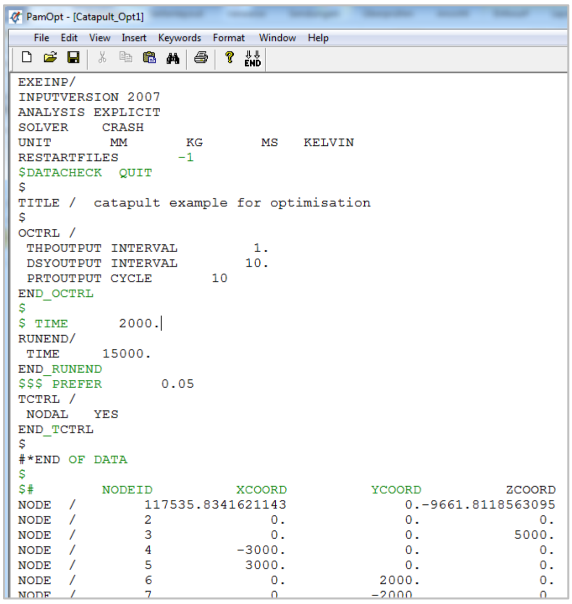
New (bottom) visiual.tpl file information
+ScriptStart
+curvexportcrv('visual.crv', ' 'p2w1c1'')

2.5 Preparing the PAM-OPT file

PAM-OPT reads a dataset that comprises of two parts; namely, a first part comprising of information to be used by PAM-OPT such as the type of optimiser to be used, the Design Parameters and the Objective Function decription. The second part comprises the standard PAM-CRASH file with possible inclusions such as so-called 'stickers'. An explanation of some of these parameters and how to setup the PAM-OPT file is given below for the catapult example.

1. Start the program PAM-OPT Editor 2005 and read in (Open) the file Catapult_Opt1.pc from the directory Catapult_OptimiseModel1_Gradient. You will be asked if you want to add the 'EXEINP / ' keyword to the top of the dataset; activate **Yes**.

2. The program will start and the new keyword will appear at the top of the PAM-CRASH dataset.



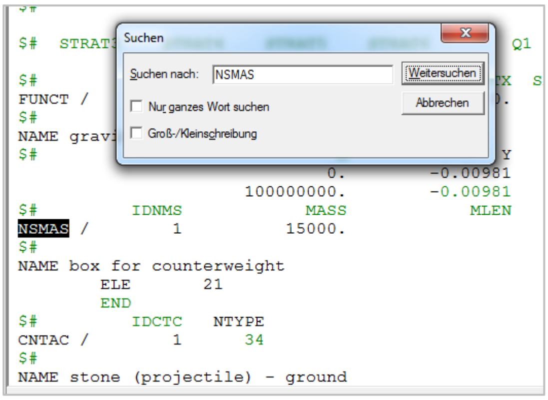
The new PAM-OPT parameters and commands will now be added to this file using the PAM-OPT editor. Most commands will be placed at the top (above the EXEINP / card) and a few (the 'stickers' are placed at precise locations in the PAM-CRASH part.

For this optimisation we shall allow the counterweight mass to be varied such that the distance to the target point is minimised. Hopefully this minimisation diistance will be zero; or very close to it.

Proceed as follows:

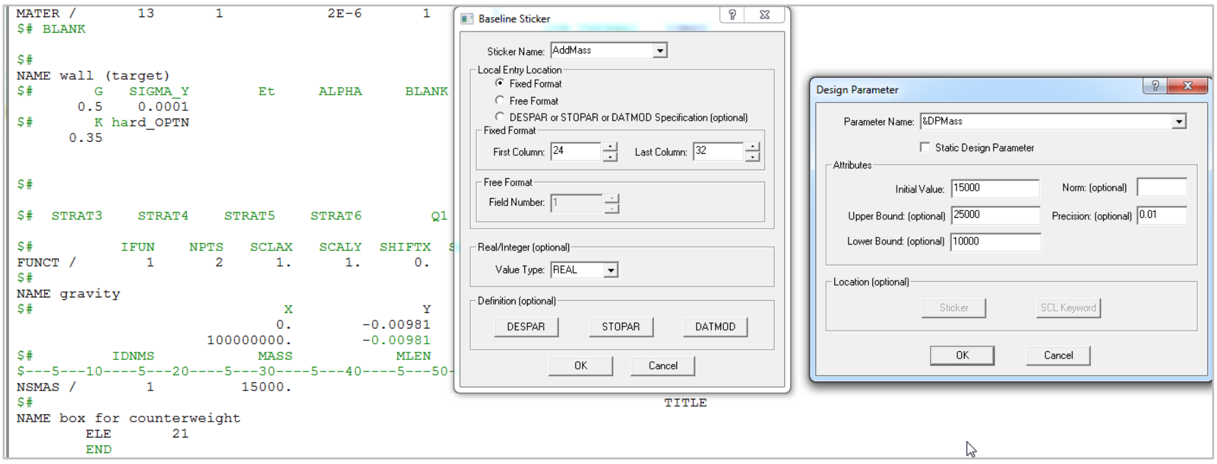
1. First we shall place a 'sticker' in the PAM-CRASH part just above the point where the counterweight mass is. This sticker will allow PAM-OPT to replace the current mass value with alternatives in the minimisation search process. Find this mass location: This is best done using Edit > Find… and giving the search keyword NSMAS.

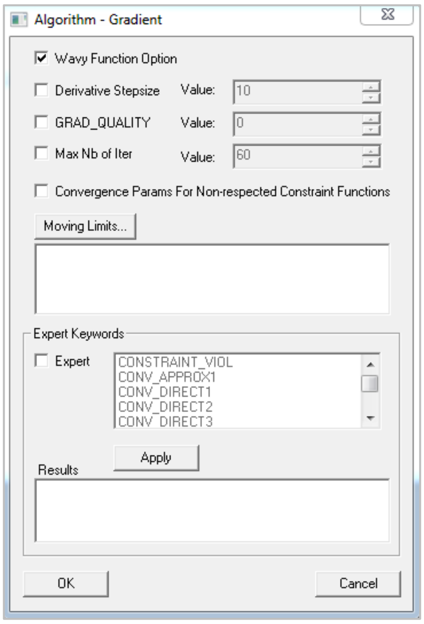
   Click on the point at the start of the NSMAS line and then activate Insert > Ruler comment. This provides useful help to locate positions for the sticker.



2. Click again on the point at the start of the NSMAS line and activate **Insert > Baseline sticker**. The first of the panels shown below will open: Give the sticker a name (e.g. AddMass), specify the location on the line where the mass is to be written (24-32), give the type as REAL and click on the **DESPAR** to open a Design Parameter panel. This is given a name (e.g. DPMass) and Initial (15000), Lower (10000) and Upper (25000) bounds for the mass are given. Finally click **OK** to the Design Parameter and Baseline sticker panels.

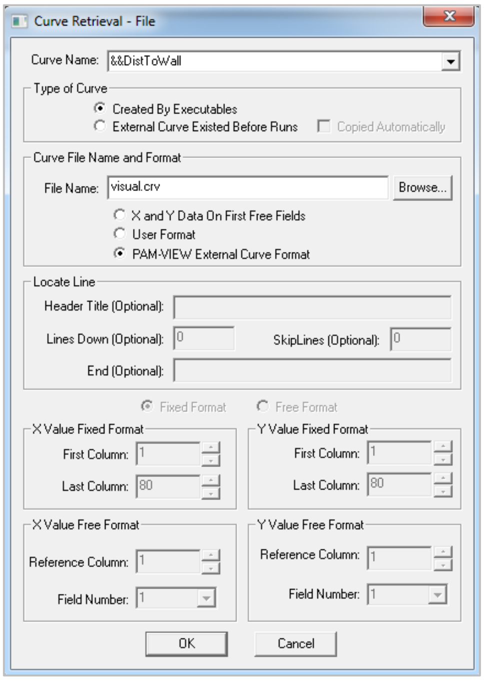
   You will fine new stick information place in the PAM-CRASH part and Design Parameter information place at the top of the dataset.





3. We shall now specify the type of optimisation analysis to be performed. Move to the very top of the dataset and click on the first position of the first line. Activate **Keywords > Algorithm** and select **Gradient**.

   Use all the defaults and close with **OK**. You will notice a **Wavy Function Option** is activated by default. This is often useful and and tries to ensure PAM-OPT looks for a global minimum and is not influenced by local oscillations in the Objective Function.



4. We shall now read in the 'Distance to wall' time history curve that will be generated in subsequent Visual-Viewer batch runs. There is no sequence to the PAM-OPT commands as long as they all appear above the 'EXEINP / ' keyword (with the exception of the 'stickers').

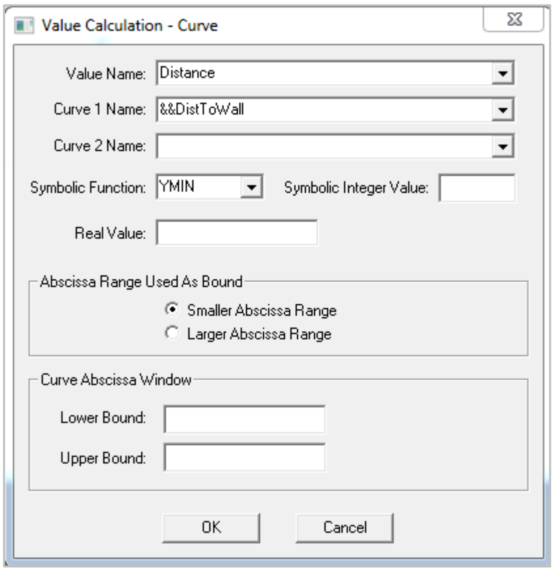
   Just above the 'EXEINP / ' keyword activate **Keywords > Curve Retrieval > File** and the adjacent panel will open. Give a suitable name (e.g. DistToWall) and give the name of the file (this was visual.crv as specified previously at the bottom of the xxx.tpl file). Activate the PAM-VIEW external Curve Format. And click **OK** to close.

   Note that this option allows many general curve types to be read and is not restricted to specific ESI results files.

5. Next we have to extract the minimim value (projectile to target point) from the curve that has just been accessed. Again, just above the 'EXEINP / ' keyword active **Keywords > Value Calculation > Curve**. The adjacent panel will open.

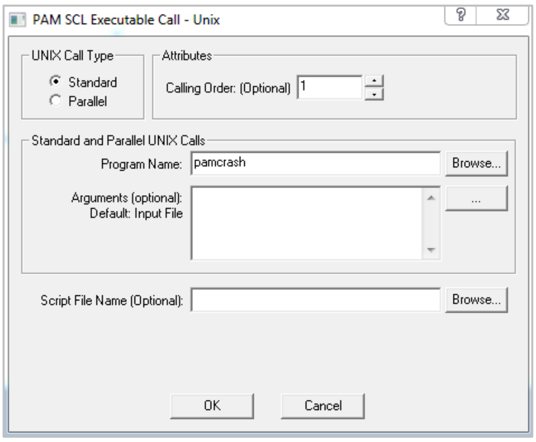
   Give a suitable name 'Distance' (this variable will eventually represent the required minimum distance result). Identify the cuve **Curve 1 Name** by clicking on the dropdown option and identifying the curve. In this case you will only see DistToWall.

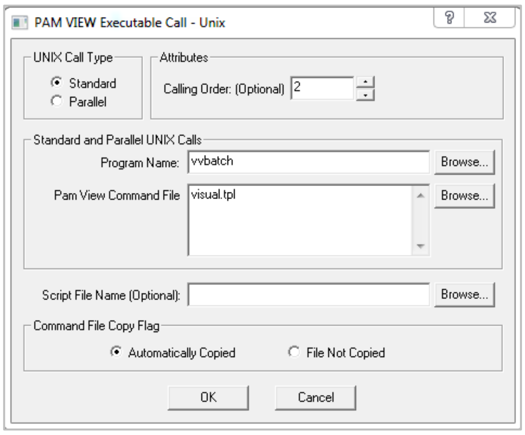
   Finally, the necessary information has to be extracted with the operation **Symbolic Function**. Click on the dropdown option and locate YMIN. Activate this to extract the minimum value from the curve and finish with **OK**.



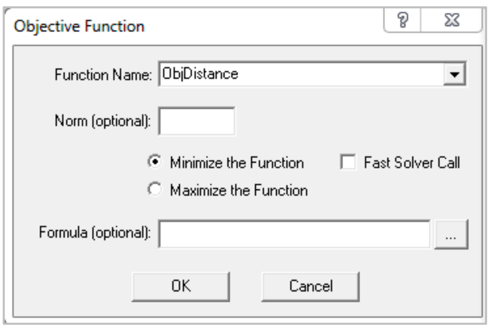
6. Now we have to define the PAM-CRASH and Visual-Viewer calls. For PAM-CRASH activate **Keywords > PAM SCL Executable Call > Unix** and specify the name **pamcrash** (see panel below); this name is recognised by the windows system and will initialise PAM-CRASH execution. It is possible to use Standard or Parallel execution depending on your computer. Check the calling order is 1 and press **OK** to finish.

   For Visual-Viewer (batch mode) activate **Keywords > PAM View Executable Call > Unix** and enter the alias name vvbatch for Program Name. The file name of the command file we established in the previous section (visual.tpl) is also specified. Check the calling order is 2 and press **OK** to Finish.

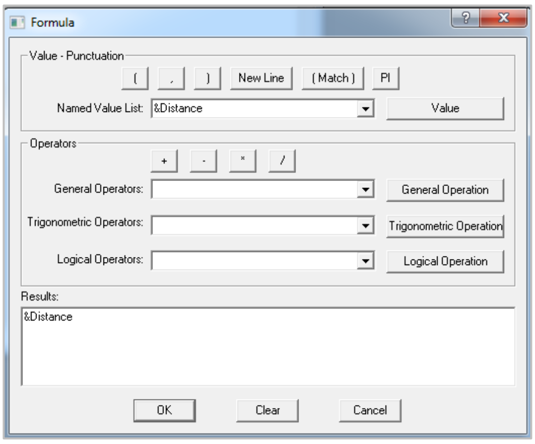




7. Finally, we have to define the Objective Function. Again, just above the 'EXEINP /' keyword activate **Keywords > Objective Function**. The first panel shown adjacent opens. Give an arbitrary name (e.g. ObjDistance), make sure the minimisation tab is active and then click **'...'** in the **Formula (optional)** line to open the second panel shown below.



In this panel complex combinations of functions, possibly mixed in formulae, can be defined. But for our simple case all we need to open is the minimum distance that we have previously found from the distance to wall time history curve. Therefore just click on the dropdown panel for Named Value List and select &Distance. Click **OK** to close both panels.



8. This completes the necessary work in the PAM-OPT editor. **Save** the dataset under a new name (e.g. **Catapult_Opt1.CDS**) and **Exit**.

9. Finally, a small change in the saved file is needed. With an editor replace VIECAL with VVBCAL to allow Visual-Viewer in batch mode to run.

10. The completed PAM-OPT **Catapult_Opt1.CDS** dataset is given below:

```
ALGKEY/ GRADIENT
      WAVY
END
$
DESPAR/ &DPMass                              0
              15000       10000       25000        0.01
      STICKER/ AddMass
$
GETCRV/ &&DistToWall                          0     0
                            visual.crv     2
$
CALVAL/ &Distance                            1
      &&DistToWall

      YMIN                                              0
$
SCLCAL/     0     1
      pamcrash

END
VVBCAL/     0     2
      vvbatch
           0      visual.tpl
$
OBJFCT/ &ObjDistance                               0     0
&Distance
END
$
EXEINP/
>>>>>>>>>>>>  Below this is the PAM-CRASH dataset with the included 'Sticker'.
```

```
$
EXEINP/
INPUTVERSION 2007
ANALYSIS EXPLICIT
SOLVER    CRASH
.......
.......
$---5---10----5---20----5---30----5---40----5---50----5---60----5---70----5---80
#STICKER/AddMass,24,32,REAL
NSMAS /        1           15000.
$#                                                          TITLE
NAME box for counterweight
        ELE       21
        END
.......
.......
#*END OF DATA
```

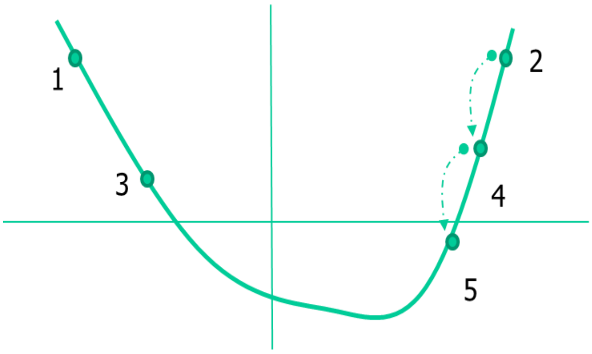
Part 3: Analysis and results for the Gradient optimisation

3.1 Brief details on the Gradient optimisation method

Gradient Optimisation: A range of conventional gradient type methods are available in PAM-OPT. In all of these the method aims to localised the bounds of the function minimum and then to work towards the minimum point. Generally, these gradient methods are efficient and require a minimum number of solver calls to locate the minimum, however, they do start to struggle if 3 or more Design Parameters have to be minimised.

In the adjacent figure points 1,2 give bounds to the minimum and point 3 identifies the minimum search direction. Points 4, 5, etc., are improved minimums obtained from the minimisation method selected.

A common problem of the method, as we shall see, is that they can locate a false minimum that does not represent the desired global minimum.



3.2 Performing the PAM-OPT analysis

Start the PAM-OPT solver and select the Catapult_Opt1.CDS input file. A listing showing progress of the optimisation should automatically appear. At the end of a successful optimisation simulation the following lines of the listing will appear which includes information on the number of iterations and solver calls (=PAM-CRASH analyses) needed to achieve the specified convergence for the optimisation (minimisation). Further information on CPU costs per simulation and the total CPU for all simulations is given.

```
 OBJECTIVE FUNCTION
  Value : 661.456
******************************************************************************
 Iteration Nb          =     1
 (SOLVER) call Nb      =     7
******************************************************************************
 PROCESS     = "exec_local.bat"

User's time ++-------------+-------------+-------------++-------------+--------+
            ||    Min      |     Max     |   average   ||    total    |  pct   |
------------++-------------+-------------+-------------++-------------+--------+
PROCESS     ||         43s |         44s |   43.714 s  ||    5mn   6s | 99.7 % |
------------++-------------+-------------+-------------++-------------+--------+
PAM-OPT     ||                                         ||          1s |  0.3 % |
Total Time  ||                                         ||    5mn   7s |  100 % |
------------++-----------------------------------------++-------------+--------+
******************************************************************************


      ============================================================
      ||   End of PAM-OPT run  (date: 2011/08/15 - 09h06m36s)   ||
      ============================================================
```

3.3 Some common problems

If the simulation fails to run successfully study carefully the PAM-OPT output listing file (listing.txt) for error messages. Typical problems could include:

- Incorrect names being used for visual.tpl, PAM-OPT and PAM-CRASH (output) files. If PAM-CRASH starts then the results are written into a sub-directory DIRINTER_xxx, where xxx is the date and time. For each new PAM-OPT run a new subdirectory is created and the last in the list corresponds to the last submission. Check that the xxx.THP and xxx.DSY file names correspond to those used in the visual.tpl and PAM-OPT dataset.
- Check the PAM-CRASH xxx.out file for successful completion. Does the last line 'NORMAL TERMINATION' appear ?
- Check in the sub-directory DIRxxx that necessary results curves (or other) information is successfully written and correct.

3.4 The results files

Once PAM-OPT is successfully started the following files and directories are created:

1. The main results of the optimisation analysis are given in the listing.txt file.
2. All PAM-CRASH analysis results are written into the new sub-directory (DIRINTER_xxx) which is automatically created each time PAM-OPT is started. The directory are conveniently cleaned by double clicking on the PAMOPT_cleanxxx file.
3. The File.doe excel results file gives summary information on evolution of the Design Parameters and Objective Function during the optimisation process.
4. The File.history and File.historyN files gives similar data to the File.doe file, but in text format.
5. The File.Curves file gives evolution of Design Parameter, Objective Function and additional information that may be plotted with Visual-Viewer.

DIRINTER_D110815H090127	15.08.2011 09:06	Dateiordner	
Catapult_Opt1	15.08.2011 07:55	CDS-Datei	93 KB
File.curves	15.08.2011 09:06	Textdokument	1 KB
File.doe	15.08.2011 09:06	Microsoft Excel 97...	1 KB
File.history	15.08.2011 09:06	Textdokument	2 KB
File.historyN	15.08.2011 09:06	Textdokument	2 KB
listing	15.08.2011 09:06	Textdokument	13 KB
PAMOPT_cleanD110815H090127	15.08.2011 09:01	Windows-Batchda...	2 KB
visual	14.08.2011 08:27	TPL-Datei	4 KB

Typical contents of the DIRINTER_xxx sub-directory are:

1. The modified PAM-CRASH dataset (xxx.inp) is created by PAM-OPT with estimated Design Parameters having been set.
2. All PAM-CRASH results files are written to this directory.
3. The visual.tpl file is copied from the upper directory for use with Visual-Viewer in batch mode.
4. The important results file from Visual-Viewer has been generated (visual.crv).
5. The remaining files are script files generated by PAM-OPT for control of PAM-CRASH and Visual-Viewer. Do not touch!

Catapult_Opt1	15.08.2011 09:01	CDS-Datei	95 KB
Catapult_Opt1.CDS.DSY	15.08.2011 09:06	DSY-Datei	46.304 KB
Catapult_Opt1.CDS.inp	15.08.2011 09:05	INP-Datei	94 KB
Catapult_Opt1.CDS	15.08.2011 09:06	Outlook-Element	2 KB
Catapult_Opt1.CDS	15.08.2011 09:06	OUT-Datei	4.669 KB
Catapult_Opt1.CDS	15.08.2011 09:06	THP-Datei	2.956 KB
interface_script	15.08.2011 09:01	Windows-Batchda...	1 KB
kill_interface	15.08.2011 09:05	Windows-Batchda...	1 KB
kill_pid	15.08.2011 09:01	Windows-Batchda...	1 KB
OS_sharedCommands	15.08.2011 09:01	Windows-Batchda...	1 KB
pamopt_interface_data	15.08.2011 09:06	Datei	616 KB
proc_input	15.08.2011 09:05	Datei	1 KB
proc_output	15.08.2011 09:06	Datei	1 KB
scriOpt0	15.08.2011 09:01	Windows-Batchda...	1 KB
scriOpt1	15.08.2011 09:01	Windows-Batchda...	1 KB
visual	15.08.2011 09:06	CRV-Datei	616 KB
visual	14.08.2011 08:27	TPL-Datei	4 KB
VisualEnvError	15.08.2011 09:06	Textdokument	2 KB

3.5 Some example results for the Gradient based optimisation

From the 'listing.txt' file progress and the number of iterations/calls needed to achieve the default convergence are given. At the end of this file the optimised counterweight mass is found to be 22564 kg with an Objective Function equal to 661 (this represents the distance to target of 0.66m).

```
System Command: "scriOpt0.bat  "

System Command: "scriOpt1.bat  "
+----------------------------------------+----------------+--------------+
|                     design parameters  |          value |  relat value |
|                               &DPMass  |        22564.5 |  1.5043      |
+----------------------------------------+----------------+--------------+
|                     objective function |          value |  relat value |
|                          &ObjDistance  |        661.456 |  661.456     |
+----------------------------------------+----------------+--------------+
 SOLVER time count = 44 sec(elapsed time)


*************************************************************************************
********************************************** GLOBAL RESULTS ***********************
*************************************************************************************
```

A further summary of this information is given in the 'file.history' file.

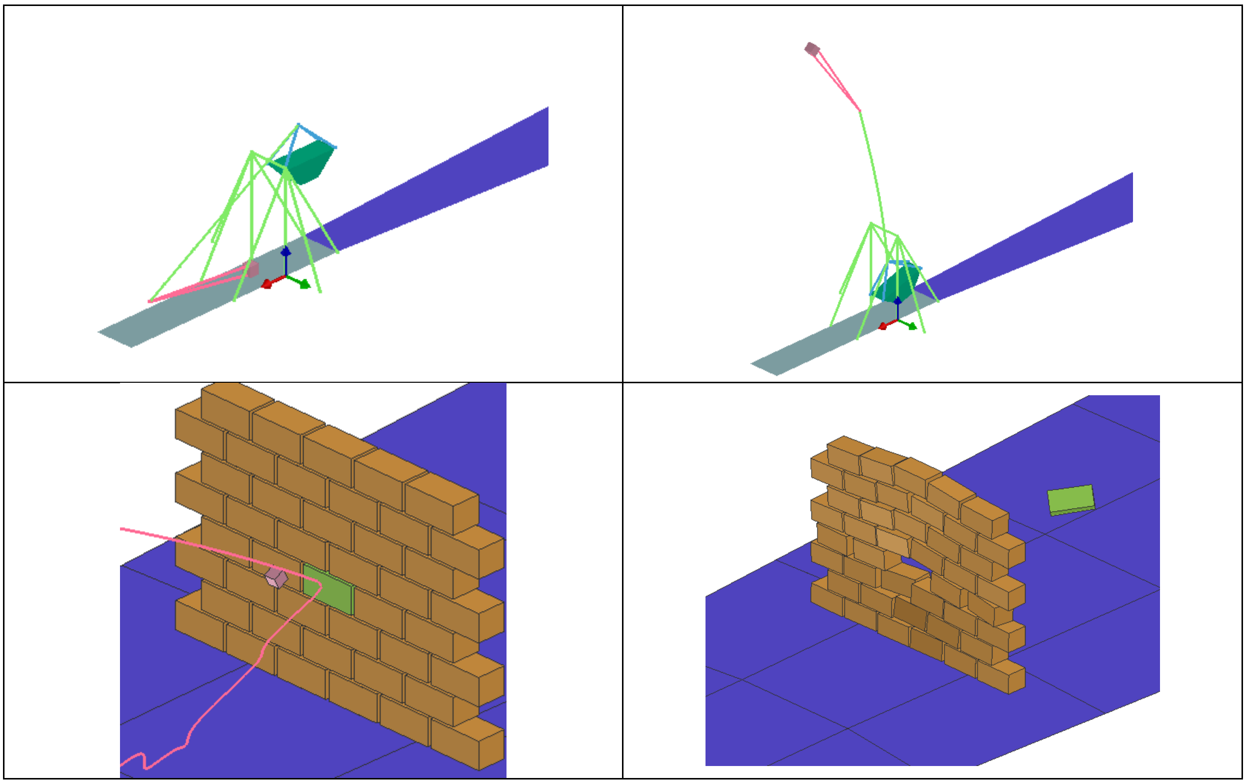
```
     +---------------------------------+
     | GRADIENT: Unconstraint Minimization |
     +---------------------------------+

*****************************************************************************
*  There is 1 design parameter :                    Lower Bound /  Upper Bound
* P1    --> &DPMass                                       10000 /        25000
*
*
*  The objective function :
* Obj   --> &ObjDistance
*
*****************************************************************************



                ****** HISTORY ******


   iter   --->        0             1
-------------|------------------------
P1:&DPMass   |  15000        22564.5
-------------|------------------------
Obj:&ObjDist |  19921.5      661.456
-------------|------------------------

  +-----------------------------+
  |   End of the PAM-OPT run    |
  +-----------------------------+
```

A concise overview of evolution of the Design Parameter (counterweight mass) and corresponding Objective Function (distance to target point) is given in the file.doe excel file shown below.

NUMBER OF PARAMETERS	
&DPMass	
NUMBER OF FUNCTIONS :	
&ObjDistance	
MATRIX	
$ P1:&DPMa	Obj:&ObjDis
15000	19921.5
16500	9852.94
25000	16538.2
23000	2816.21
20188	11504.3
22341	2089.72
22564.5	661.456

The best (or close to the best) results are stored in the DIRINTER_xxx directory. To be sure it may be wise to rerun the problem as a normal PAM-CRASH analysis with the optimised design parameters.

From Visual-Viewer and opening the xxx.DSY file (in the DIRINTER_xxx directory), the following views and results are found. In the listing.txt file it is seen that the solution determines a counterweight mass of 22,565 kg and a good impact distance to the target point of 0.66m (Objective Function = 661).



Useful Exercise:
Try increasing the level of convergence to force more optimisation solutions. This is done by modifying the paramener in the DESPAR card for the mass Design Parameter (default = 0.01). Try, for example, 0.0001 and compare results (CPU, interations, counterweight mass and Objective Function) with the previous results found above.

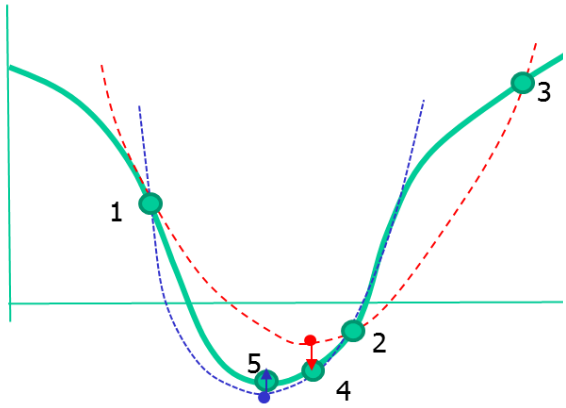
Part 4: Other optimisation methods (RSM and Genetic)

4.1 Organisation of files

In this part the previous catapult model optimised using the Gradient method will be re-analysed using the RSM (Response Surface Method) and Evolutionary (Genetic) optimisation methods. For each method:

1. Create a new directory for each optimisation method (e.g. Catapult_OptimiseModel1_RSM and Catapult_OptimiseModel1_Genetic).
2. Copy the files Catapult_Opt1.CDS and visual.tpl from the Gradient directory to each of these directories.

4.2 Brief details on each optimisation method (RMS type and Genetic)



RSM Optimisation: The principles of the method are comparable to Parabolic interpolation and Brent's method in one dimension[1]. Note PAM-OPT uses different methods based on 'adaptive bounds parameters' and Brent's method is explained here just to illustrate the concepts.

Essentially design points 1, 2 and 3 are computed to obtain their Objective Functions. A parabola is fitted through these points and its numerical minimum at 4 is used to compute a new Objective Function. From this a new parabola is placed through updated points 1, 4 and 2 and the process is repeated for point 5. This repeats until a convergence criteria is reached.

The general RMS method can be applied to 1, 2, or more, design variables. Here the Objective Function is computed at a specific network of points over the design space. From these points a higher order polynomial (response) surface is fitted to approximate the true Objective Function surface. The higher the order of the response surface the better will be the fit, but this will become computationally expensive. For example a third order polynomial with 3 design variables would require 3^3 points (analyses) to fit the function; whereas a second order polynomial would require only 3^2 points; consequently, a second order polynomial is usually preferred.

Based on the results the design domain can be narrowed giving an updated region of interest. Within this region a new network of points are computed and the response surface updated. As in Brent's method the process converges toward a minimum.

Genetic Optimisation: The Genetic method is based on concepts of evolution of a population involving selection of the fittest (best solutions), combining (mating) the fittest of these solutions and occasional mutation to add new potential solutions. The method does require many analyses, but can be very effective to overcome difficulties of entrapment at local minimums, which is often a problem encountered with conventional Gradient and similar methods (e.g. RSM).

[1] "Numerical Recipes: The art of Scientific computing", Press, Flannery, Teukolsky and Vetterling, Cambridge University Press.

Very briefly the Genetic method replaces the decimal representation of the Design Parameters with a binary representation. For example 14kg is defined as 1110 ($=1*2^3 + 1*2^2 + 1*2^1 + 0$). A population of user defined size is determined over the design range; in this case these are random masses for the counterweight mass between 10,000kg and 25,000kg which are expressed in binary format. For each of the population a simulation is performed and a selection process identifies to 'fittest' solutions. The best of these, possibly with duplication, are carried through to the next population and the worst are eliminated.

A crossover operation is then performed. This partners pairs of solutions (randomly)and information is exchanged. In this case the information exchange is a crossover operation in which bits are exchanged at a random points in the binary string. A small percentage of mutation is added in which further random bit exchanges are swopped (1 to 0 or vice versa). The new population is examined and generally better (fitter) solutions are found. The method is then repeated until a convergence is reached, or user defined limits to the operations are reached.

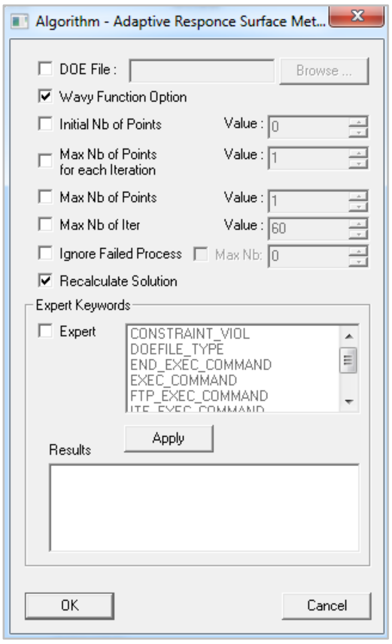
Note that the essential difference of this method is that the best potential solutions over the design space are identified and brought forward for modification and further examination. This is fundamentally different to Gradient and RSM type methods which try to identify a minimum location and then chase down the minimum value.

4.3 RSM Optimisation and results

1. Start the PAM-OPT editor and read in the Catapult_Opt1.CDS from the Catapult_OptimiseModel1_RSM directory.

2. Delete the cards:

```
ALGKEY/ GRADIENT
         WAVY
END
```



3. Activate **View > Refresh Lists** to invoke this deletion.

4. Now specify the new optimiser type; again, this is best located at the top of the dataset. Move to the very top of the dataset and click on the first position of the first line. Activate **Keywords > Algorithm** and select **Adaptive RSM**.

   Use all of the defaults but do also active the '**Recalculate Solution**' option. This causes the best solution at the end of the optimisation process to be recomputed for inspection.

   Again, use the **Wavy Function option** and close with **OK**.

4 **Save** (overwrite) the modified dataset and **Exit** the PAM-OPT editor.

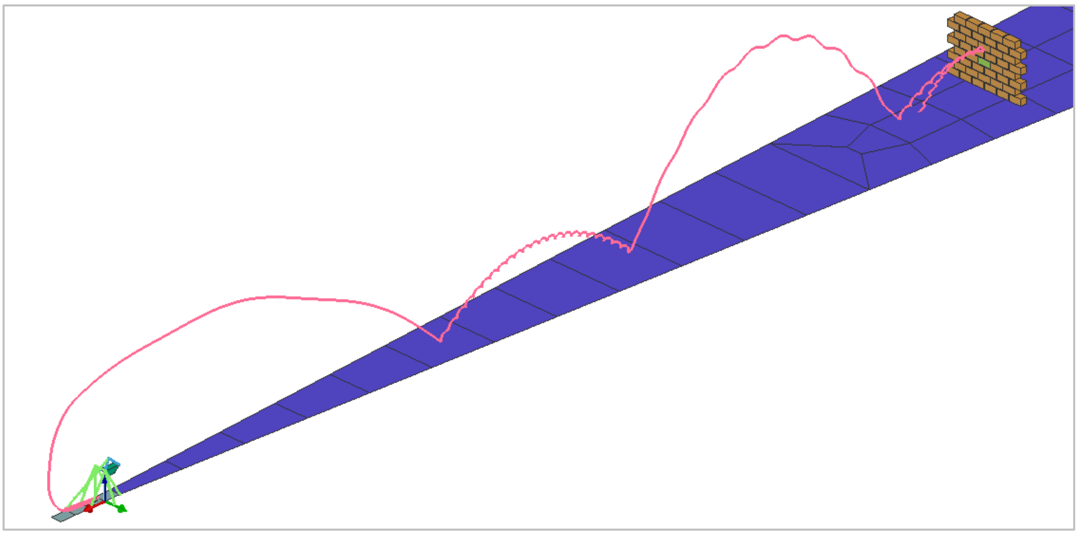
The results are slightly better (ca. 20%) to the previous Gradient method results; but are, in this case, computationslly more expensive (+50%).

Method	**CPU**	**Mass**	**Objective Function**	**Result**
Gradient	5 mins (7 analyses)	22564	661	Good - on target
RMS	7.5 mins (10 analyses)	22575	502	Good - on target

4.4 Genetic Optimisation and results

Identical operations to those in section 4.3 are performed to set-up the Genetic optimisation model. For this case use a Population size of 5 and Inter_NB size of 5. Note these values are very small, but used here to keep the CPU costs low. Typically population sizes are 20-30, or higher (PAM-OPT default = 100), but this largely depends on the problem treated and CPU available.

The Genetic result shown below show that the projectile did not manage to give a direct hit to the target, but was vary close. In this case a number of bounces occurred along the ground before hitting the wall. It is important to note that these results are not wrong! They have found a minimum (= valid solution) for the specified Objective Function and are equally correct compared to the previous Gradient and RMS results.



In the following further genetic optimisation analyses have been run to investigate the influence of the Population and Nb_Inter parameters

Method	**CPU**	**Mass**	**Objective Function**	**Result**
Genetic (Population =5, ITER_NB=5)	8.5 mins (12 analyses)	13500	2251	2.251 m above target – (3 bounces)
Genetic (Population =10, ITER_NB=5)	54 mins (73 analyses)	14650	1088	1.088 m below-left of target (2 bounces)
Genetic (Population =20, ITER_NB = 5)	2.25 hrs (189 analyses)	14650	1088	1.088 m below-left of target (2 bounces)

Note:

1. The Genetic method uses computer generated psudo-random numbers for the population, cross-over and mutations. Therefore you probably will not get the same results if run on different machines. The problems is made further 'irratic' by the nature of the square projectile that bounces like an 'American football'; but this does add an interesting 'additional' challenge to the optimisation.
2. Using a larger population is usually more important than the number of interations.
3. Possibly, if the Gradient or RSM method had started to localise at a minimum with bounces (which is an equally valid solution) they would have struggled to find a good result. In this respect the Genetic method probably did very well.

Part 5: Further studies: Discrete analysis

5.1 Discrete analysis

It is clear from the previous optimisation studies that for the spacified Objective Function there are many potential solutions for the projectile to hit the target point; since cases involving bouncing prior to impact are equally valid. It is possible with the **ALGKEY > DISCRETE** to perform specific discrete analyses to get Objective Function values for a set of design parameter (counterweight mass) values. This is a techniques that could investigate areas of possible minimums.

The steps to perform this type of discrete analysis are:

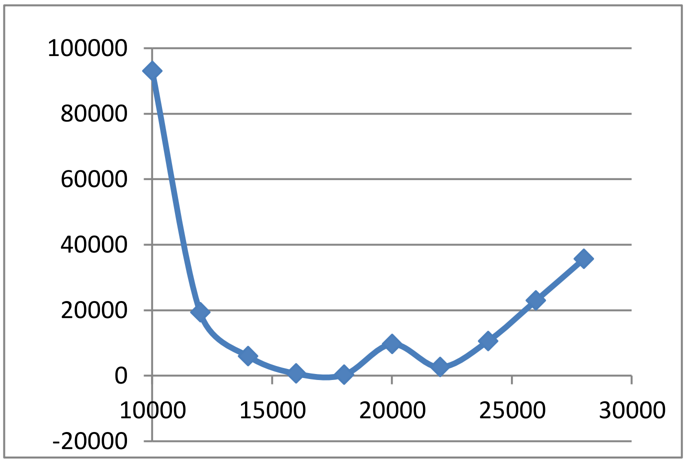
1. Create a new diectory (e.g. Catapult_OptimiseModel1_Discrete) and copy a xxx.CDS and visual.tpl file to it. We shall use Genetic optimisation so select the files used previously in Section 4.4.
2. Start PAM-OPT editor and read in the xxx.CDS file.
3. Delete the current **ALGKEY** cards and activate **View > Refresh Lists**.
4. Define and new algorithm key **ALGKEY > DISCRETE**. The Design Parameter(s) is selected (&DPMASS) and then the set of discrete values (shown below) are given; note each value is given independently and the list is updated with **Apply**.
5. Finally, **Save** and **Exit**.

The following information and input cards were created for this example,

```
ALGKEY/ DISCRETE
        &DPMass ; 10000 12000 14000 16000 18000 20000 22000 24000 26000
28000 30000
        RECAL_SOLUTION
END
```

From this a PAM-OPT analysis will perform 11 analyses. The adjacent graph is generated from the results xxx.doe excel file and gives variation of Objective Function and corresponding counterweight mass.

The discretisation is coarse, but possible minimums in the range 13000-18000 and again in the range 21000-23000 are evident. These correspond to the regions of actual minimums so far identified with the Gradient, RMS and Genetic algorithms.



Note that the process could be repeated with a refined list of counterweight masses around the areas of identified minimums. This would be a manual approach to minimize a function.

Part 6: Further studies: Constraints

6.1 Constraints

An important aspect of optimisation may be to converge to a specific solution that also ensures certain other conditions are met; such as avoiding ground contact before hitting the target. Various methods are possible to constrain against this; for example:

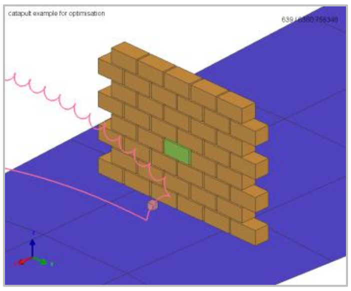
1. The counterweight mass limits defined in the Design Parameter could be increased to avoid early ground contact.
2. The Objective Function could be modified to include additional information that would try to maximise counterweight mass, while still aiming for the target point. Clearly the case of maximum counterweight mass should avoid ground contact.
3. An additional constraint function can be imposed; also, with the aim to maximise the counterweight mass.

For these studies the genetic optimiser is used as this seems always prefer solutions with low mass and ground contacts prior to impact.

6.1.1 Constraint: Via mass limits in the Design Parameters

For this case the counterweight mass values in the Design Parameter definition are increased to have limits that avoid ground contact before hiiting the target. The steps to perform this are:

1. Create a new diectory (e.g. Catapult_OptimiseModel1_Genetic_Constraint_DPHeavymass) and copy to it (from the Section 4.4 work) the xxx.CDS and visual.tpl files.
2. Either via PAM-OPT editor, or any other editor, modify the masses on the DESPAR card having start, lower bound and upper bound values of 25,000, 20,000 and 30,000 respectively.
3. Save the dataset and rerun the problem with PAM-OPT.



Result: You should see that a soulution has been sought that gives a couterweight mass of ca. 21980 kg and an impact distance of ca. 1.8m (Objective Function = 1769). Again, the Genetic opimiser has found a solution with one short bounce before wall contact (which is valid). Note due to the random nature of the Genetic algoritm you may get different results. Also, a population of only 5, used for tutorial purposes, is really too small; 20-30 would be better.

6.1.2 Constraint: Via a modified Objective Function

Creating an Objective Function that tries to meet the distance criteria (projectile to target point) and maximise the counterweight mass to avoid bounces on the ground is not strightforward. Some experimentation is usually needed to get the right Objective Function for this type of prolem. Note that trying to minimise these two inpendent parameters, distance and mass, in the Objective Function is a case of multi-objective minimisation.

Proceed as follows:

1. Make a new directory (e.g. Catapult_OptimiseModel1_Genetic_Constraint_ ObjFunctionMass).
2. Copy the xxx.CDS and visual.tpl files (from Section 4.4 work) to this directory.
3. Start PAM-OPT editor and load the xxx.CDS file.
4. Delete the old Objective Function and activate **Views > Refresh List.**

5. Now a new Objective Function can be defined with the generated cards shown below. The choice made here is to minimise target distance (&Distance) and weight this distance with a factor 1./&DPMASS. The latter term is smaller, for a greater mass, giving the Objective Function a clearer minimum for the case of a larger counterweight mass.

6. The modified dataset is saved and run with PAM-OPT.

```
$
OBJFCT/ &OFDistAndMass                                                      0         0
 &Distance   * ( 1. / &DPMass )
END
$
```

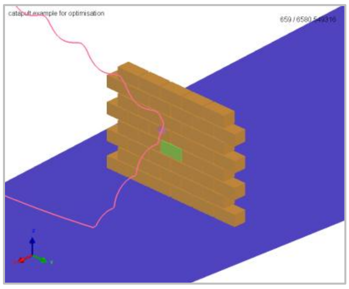
Result: The result is not particularly good with too low a mass (13500 kg); the .DSY file shows projectile contact with the ground well before contact with the target.

The Objective Function can be further improved. For example by magnifying the influence of the mass term (1./&DPMASS) either be scaling (e.g. a multiplication *10), of more severly by squareing (^2). The latter is used giving the new Objective Function shown below.

```
$
OBJFCT/ &OFDistAndMass                                                      0         0
 &Distance   * ( 1. / &DPMass ) ^ 2
END
$
```

Result: This results are still not particularly good and a greater population size is needed.

A Population of 20 (INTER_NB = 5) gives good results with a counterweight mass of (21625 kg) and only a short bounce just before wall contact.



Note: Approaches using constraint functions are usually better than complex Objective functions.

6.1.3 Constraint: Via an additional Constraint Functions

Different types of constraints could be imposed. For example a constraint could be envisaged that monitors the projectile height from release to the wall and tries to enforce a condition that no contact occurs prior to impact with the wall; this would would require additional outputs from Visual-Viewer and more data handling in PAM-OPT. An alternative, simple approach, is used here to constrain the mass to a high value, whilst also trying to meet the target point Objective Function criteria.

As in the previous sub-sections create a new directory (e.g. Catapult_OptimiseModel1 _Genetic_Constraint_ ConstFuncMass) and copy to this the xxx.CDS and visual.tpl files from the previous work in Section 4.4. Start the PAM-OPT editor and load the xxx.CDS file. To define a new constraint proceed as follows:

1. Position the cursor at the place you would like to define the constraint; e.g. just below the DESPAR keywords.

2. Activate the constraint with **Keywords > Constraint Function**; give it a name, for example, ConstMass and define high limits such as 25000 and 50000. These limits are too high and PAM-OPT will violate these in trying to minimise the Objective Function; however, they will have their effect and cause a solution to be sought with as high as possible counterweight mass.

3. Under the **Formula (optional)** and **Named Value List** select the mass Design Parameter &DPMASS. Finally, click **OK** to finish. The following cards should appear. Save and run the dataset as usual.

```
CSTFCT/ &ConstMass                                                    0
                   25000            50000
&DPMass
END
```

Result: The solution manages to find a good counterweight mass of 22000 kg and a respectable impact distance from the target of ca. 2.6m (Objective Function = 2605). Note the mass in the constraint conditions has been violated but it has effectively imposed the required condition to maximise the counterweight mass value.

Part 7: 2D and multi-objective optimisation

7.1 The study

The previous studies were 1D (one dimensional) optimisation as they only involved one Design Parameter. Usually optimisation studies involve 2, 3 or more Design Parameters that must be simulataneously adjusted to minimise the Objective Function. Furthermore, the Objective Function may have single, or multiple parameters, to be minimised.

In order to illustrate this the previous study is extended to have two design parameters; namely,

1. The current counterweight mass Design Parameter is maintained.
2. A new Design Parameters for release position of the projectile is introduced. Release is controlled via distance of the projectile from a fixed floor point in a PAM-CRASH sensor card.

Further complexity is added to the Objective Function:

1. The existing criteria for distance to the target point is kept.
2. An additional criteria to force the projectile to reach a specific height is added.

Very briefly the new model requires the following steps to be undertaken:

1. A new directory is created (e.g. Catapult_OptimiseModel1_RMS_2DesignParameters) and the RMS datasets (.CDS and visual.tpl) from Section 4.3 are used.
2. A new bar elements is added to the PAM-CRASH part of the dataset (e.g. set at a height such as 65,000mm) to provide a visual reference for the projectile height (this is optional).
3. The existing criteria for distance to the target point is kept and additional information is provided (time history curve) to monitor height of the projectile. The new time history is z-coordinate of the projectile. For simplicity a second Visual-Viewer run was made in batch mode to generate this new new curve. In the following example the two visual batch xxx.tpl files are visualTG.tpl (this is the same as the previous visual.tpl) and visualHT.tpl (this is new and has to be generated).
4. A new 'Sticker' (Release) and associated Design Parameter (DESPAR card) is added to control the release point of the projectile from the sling.
5. An additional criteria to force the projectile to reach a specific height (65,000 mm) is added. This requires opening the new curve in PAM-OPT, extracting the maximum height (YMAX) and adding this height (relative to the required height) to the Objective Function expression.

The following extracts of the completed datset illustrate the new input:

<u>Tutorial 14: General exercise in optimisation using PAM-OPT</u>

Tutorial 14: General exercise in optimisation using PAM-OPT

For the main PAM-OPT parts of the input

```
ALGKEY/ ADAPTIVE_RSM
        WAVY
        RECAL_SOLUTION
END
$
DESPAR/ &DPMass                                       0
                   25000          10000          50000           0.01
        STICKER/ AddMass
$
DESPAR/ &DPRelease                                    0
                   18000          14000          20000           0.01
        STICKER/ Release
$
GETCRV/ &&DistToWall                                  0       0
                            visualTG.crv              2
$
GETCRV/ &&Height                                      0       0
                            visualHT.crv              2
$
CALVAL/ &DistTG                                       1
        &&DistToWall

        YMIN                                                                  0
$
CALVAL/ &DistHT                                       1
        &&Height

        YMAX
$
SCLCAL/        0       1
        pamcrash

END
VVBCAL/        0       2
        vvbatch
               0        visualTG.tpl
$
VVBCAL/        0       3
        vvbatch
               0        visualHT.tpl
$
OBJFCT/ &ObjDist                                              0       0
&DistTG + ABS ( 65000.0 - &DistHT ) |
END
$
EXEINP/
INPUTVERSION 2007
ANALYSIS EXPLICIT
SOLVER    CRASH
```

For the main PAM-CRASH parts ('Stickers' only)

```
$#          IDNMS              MASS              MLEN              MARE              MVOL
$---5---10----5---20----5---30----5---40----5---50----5---60----5---70----5---80
#STICKER/AddMass,24,32,REAL
NSMAS /         1           15000.
$#                                                                              TITLE
NAME box for counterweight
        ELE         21
        END
```

```
NAME displacement sensor for release (distance between nodes)
$# BLANK       N1       N2      INDT      INDC     SIGLN     SIGSH   MOTION
$---5---10----5---20----5---30----5---40----5---50----5---60----5---70----5---80
#STICKER/Release,42,48,REAL
            2005         5         1         0  18000.         0.       10
```

Result: Generally, the results shown below are encouraging, but some improvements could be made. Usually alternative optimisation methods, more critical convergence criteria, closer design limits, or improved Objective Function are some of the methods used to improve results.

The two cases are considered below use alternative Objective Functions

1. &DistTG + ABS (65000.0 - &DistHT)
2. &DistTG ^ 2 + ABS (65000.0 - &DistHT) ^ 2

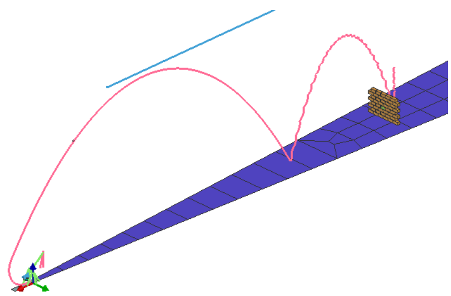
The second case can sometimes be useful to exaggerate Objective Function numbers around the minimum point and help the minimisation process. However, this has led to the interesting results below, which illustrates the need to get the Objective Function right! In this case the top of the wall was hit and led to a second bounce that got quite close to reaching the height objective. <u>This is a valid solution for the Objective Function specified</u>. In hind sight a better height criteria would be to extract height from a cross plot of height (z) versus length (x) coordinates for the projectile, between limits of release point to the target.

Case 1

&DistTG + ABS (65000.0 - &DistHT)

CPU ca. 22 mins

```
----------------------+---------------
 design parameters    |          value
           &DPMass    |        20482.1
        &DPRelease    |        15281.4
----------------------+---------------
 objective function   |          value
          &objDist    |        7059.57
----------------------+---------------
```

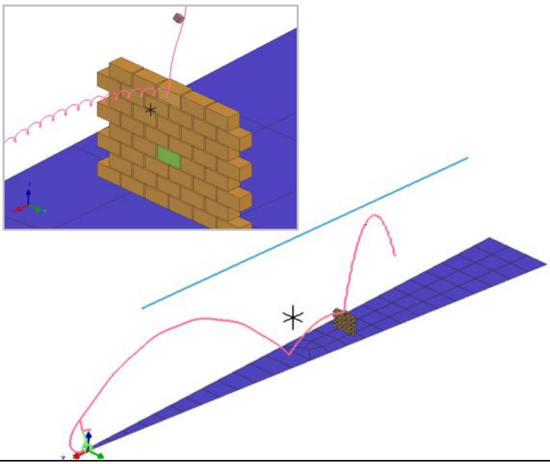


Case 2

&DistTG ^ 2 + ABS (65000.0 - &DistHT) ^ 2

CPU ca. 22 mins

```
----------------------+---------------
 design parameters    |          value
           &DPMass    |        18467.4
        &DPRelease    |        16633.6
----------------------+---------------
 objective function   |          value
          &objDist    |   8.93023e+007
----------------------+---------------
```



Part 8: The final optimised model

With a few changes a final (optimised model) is provided in the directory xxx_Final. Note that this is a standard xxx.pc file to be run with PAM-CRASH (not with PAM-OPT).