Parameterised Analysis Model for a Quick Opening Valve in an Idealised Single Stage Gas Gun

Dean J. Dvornicich[1]

University of New South Wales at the Australian Defence Force Academy

Gas gun systems generally consist of a high pressure gas reservoir connected by a quick opening valve to a projectile chamber. High acceleration of valve components make valve dynamics and stress analysis an essential component of gas gun design. A model has been created utilising various software packages that allow a user to easily model an idealised quick opening valve within a single stage gas gun. The model automatically performs dynamic analysis, fluid mechanic analysis, three dimensional model and mesh creation, and finite element analysis. The model requires only a set of variables describing the valve to operate. The model allows a user to analyse the effect of varying multiple parameters within a valve design quickly with only a basic knowledge of Matlab encoding.

Contents

Nomenclature

m	=	mass [kg]	R	=	specific gas constant [$JK^{-1}mol^{-1}$]
v	=	velocity [ms^{-1}]	T	=	temperature [K]
t	=	time [s]	V	=	volume [m^3]
F	=	force [N]	k	=	ratio of specific heats
$\ddot{x}$	=	acceleration [ms^{-2}]	Ma	=	mach number
P	=	pressure [Pa]	c	=	speed of sound [ms^{-1}]
A	=	area [m^2]	P^*	=	critical pressure [Pa]
ρ	=	density [kgm^{-3}]	$\dot{m}$	=	mass flow rate [kgs^{-1}]

[1] SBLT, School of Engineering & Information Technology. ZEIT4501

I. Introduction

Flow control valves are a system component used in fluid and mechanical systems to adjust variables such as mass flow rate, pressure or temperature (Engineering Design, 2011). Gas gun systems generally consist of a high pressure gas reservoir connected by a quick opening valve to a projectile chamber. High acceleration of valve components make valve dynamics and stress analysis an essential component of gas gun design. This project provides a parameterised analytical tool in which an idealised single stage piston valve in a gas gun can be quickly and easily analysed by a user with little experience in computational analysis software.

Software utilised in this project includes Matlab, Simulink and Abaqus. A user is required to provide a set of variables describing the valve system they wish to model. The software tool will then return data including valve and projectile dynamics, as well as fluid mechanic properties such as mass flow rate and system pressures. The tool will also automatically generate a three dimensional model of the valve piston in Abaqus and perform Finite Element Analysis (FEA) on the piston component.

The tool is designed to offer a high flexibility in the level of skill of the user in Matlab, Simulink and Abaqus. At the most basic level, a user can do as little as edit a script file to contain their valve variables, then run an included user interface that will load the variables for them, run the simulation, model the piston in Abaqus, and perform FEA automatically. At more advanced levels, a user may prefer to write a script that utilises the in built functions to alter one or more variables, perform chosen analysis, and record the results for many different configurations.

This report describes the design and use of the software tool, including the assumed system configuration, fundamental principles applied, and the modular software design. Examples of use of the software tool are also included.

II. Valve System Configuration

The software tool assumes the valve system is a single stage piston valve configuration as shown in Figure 1. In this system, fluid is pressurised in a large reservoir. The piston is actuated at a given initial velocity in the direction shown. Once the face of the piston moves past the chamber orifice gas begins to flow from the reservoir to the chamber. Acceleration of the piston and projectile then occurs due to the pressure in the chamber. In this model, the projectile is assumed to exit to atmosphere after a given chamber length. The piston is obstructed by a sealed cushion of gas on the face of the piston opposite to the chamber.

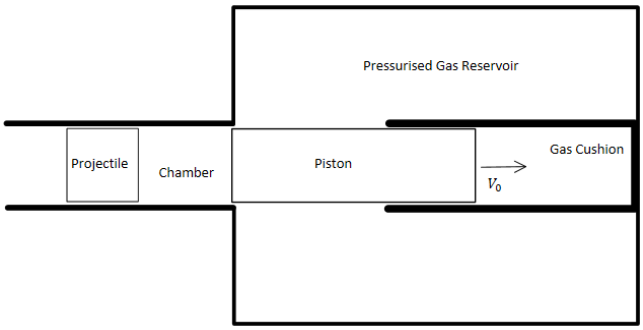


Figure 1 - Valve System Configuration

The piston is assumed to be cylindrical with a small chamfer radius on the outer cylinder edges. Piston and projectile material is assumed to be homogeneous with a defined density, Poisson's ratio, and modulus of elasticity. The piston can be defined as solid or as having a cylindrical hollow section.

A full set of required user input variables, their descriptions, and the values used in examples is included in Appendix A.

III. Fundamental Physical Principles

The model calculates the idealised piston and projectile dynamics properties as well as the fluid flow properties. Piston and projectile dynamics are dependent on the components mass and the pressure across the surface of the components (Hibbeler, 2010). Chamber side surface pressure is dependent on the properties of the fluid, the fluid flow area, and the geometry of the piston, projectile, chamber and cushion (Munson, et al., 2010). As the fluid flow area is also a function of the piston dynamics, the interaction between the piston and fluid flow forms a complex problem, which is solved using numerical methods in Simulink and Matlab. The fundamental physical principles governing the dynamics and fluid mechanics in the model are described in this section.

The model is an idealisation of a real system. As such, the following assumptions are made in the model:

1) Gas behavior is considered ideal.
2) Gas expansion and compression is considered isentropic.
3) Flow is considered adiabatic and frictionless. A mass flow rate coefficient is used to compensate for non-ideal effects as is industry standard practice for valve systems (Headley, 2003).
4) The pressure profile across the surface of the piston and projectile is averaged to the mean pressure in the chamber during initial fluid flow.
5) Friction throughout the system is ignored.

Details of these assumptions and effects of them is discussed in the relevant proceeding sections.

A. Piston and Projectile Dynamics

Piston and projectile dynamics are governed by Newton's second law of motion (Hibbeler, 2010). This law applied to the piston and projectile can be seen in equations 1.

$$\frac{d(mv)}{dt} = \sum F$$

$$\therefore m_{piston}\ddot{x}_{piston} = \int P_{chamber} \cdot dA_{piston} - \int P_{cushion} \cdot dA_{piston} \qquad (1)$$

$$m_{proj}\ddot{x}_{proj} = \int P_{atm} \cdot dA_{proj} - \int P_{chamber} \cdot dA_{proj}$$

Source: **(Hibbeler, 2010)**

As discussed previously, pressure across the chamber side surface of the piston and projectile are averaged to the pressure in the chamber. This assumption holds true when the velocity of the fluid across the surfaces is zero; that is once chamber pressure has reached equilibrium pressure with the reservoir. During initial flow conditions a pressure gradient will exist, particularly across the surface of the piston (Munson, et al., 2010). This effect is considered negligible in the context of this model. Friction is also ignored in this model, however it is relatively simple to add a friction component to the model if necessary in a particular design.

B. Fluid Mechanics

Fluid compression and expansion is calculated through the ideal gas law and isentropic relations shown in equations 2.

$$P = \rho RT \qquad PV^k = constant \qquad \frac{T_2}{T_1} = \left(\frac{P_2}{P_1}\right)^{\frac{k-1}{k}} \qquad \frac{T_2}{T_1} = \left(\frac{V_2}{V_1}\right)^{1-k} \qquad (2)$$

Source: **(Cengel, et al., 2012)**

Fluid flow properties are approximated using a model of compressible isentropic flow through a converging nozzle. A user defined coefficient compensates for non-ideal effects. Velocity and fluid property changes are considered to occur in the streamwise direction only. Equations 3 are derived from Newton's second law of motion applied to inviscid flow and conservation of mass. Detailed derivations are beyond the scope of this report, but are available in (Munson, et al., 2010). Subscript zero denotes stagnation values.

$$v = Ma \cdot c = Ma\sqrt{RTk} \qquad \dot{m} = \rho Av \qquad \frac{P^*}{P_0} = \left(\frac{2}{K+1}\right)^{\frac{k}{k-1}}$$

$$\frac{T}{T_0} = \frac{1}{1+\frac{1}{2}(k-1)Ma^2} \qquad \frac{P}{P_0} = \left[\frac{1}{1+\frac{1}{2}(k-1)Ma^2}\right]^{\frac{k}{k-1}} \qquad \frac{\rho}{\rho_0} = \left[\frac{1}{1+\frac{1}{2}(k-1)Ma^2}\right]^{\frac{1}{k-1}} \qquad (3)$$

The model determines when flow is choked and adjusts calculations automatically. Choked flow occurs when pressure in the minimum cross section flow area is less than the critical pressure of the fluid. The Mach number is defined as 1 for choked flow. Flow properties are only significant during the initial opening of the valve, prior to the chamber pressure equalizing to the reservoir pressure. Depending on the system, this is typically a short period and often considered negligible in initial design calculations (Seidl, et al., 2013).

IV. Project Design

A modular design approach has been used in creating the system model. Each module handles a separate component of the analysis. A chart describing each module and their interaction is included in Figure 2.

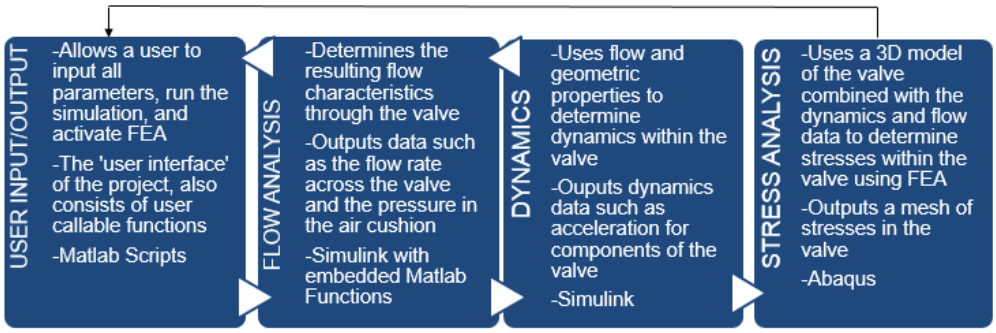


Figure 2 – Modular System Design

Flow Analysis and Dynamic Analysis modules are contained in a Simulink model, an image of which can be seen in Figure 3. A full print of all Simulink model modules is attached in Appendix B. The Simulink model is also capable of showing an animation of the valve piston and projectile. FEA is conducted in Abaqus either with or without the Abaqus Graphical User Interface (GUI). User input and output as well as data interaction between programs is achieved through Matlab scripts and functions. The following sections describe each module's function and capability.

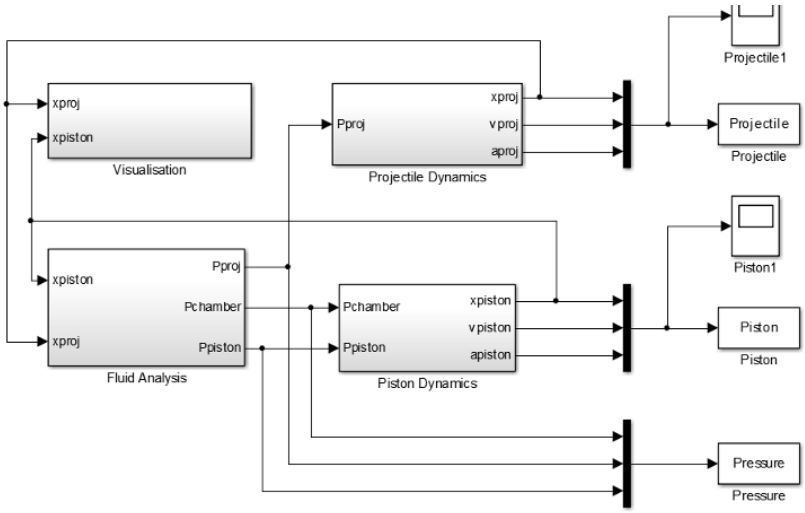


Figure 3 – Highest Level of the Simulink Model

A. User Interface Module

This module consists of a user interface script, a script for variable storage, and various functions that the user can use to perform simulations and FEA. The user interface script allows a user to set required variables, run a simulation, generate an Abaqus model, and submit a job for FEA in Abaqus with or without the GUI. This can all be done without leaving the Matlab command interface. The script achieves this by prompting the user for information on what task they would like to perform, then calling required functions to perform those tasks. The user interface script is designed to be fast and easy to use in generating a single valve model. Example one in the model usage section of this report shows the user interface script and what it produces.

As an alternative to using the user interface script, a user may choose to call directly on the model scripts and functions. A user can call on a script to set the required variables for a simulation, run the simulation, and call one of two functions to automatically generate the piston model and perform FEA with or without the Abaqus GUI. Both functions automatically save the model and FEA results in a subfolder specified as in input to the function. Table 1 briefly describes each function or script and its role. The full contents of all functions and scripts are included in Appendix C.

The interface functions are particularly useful for a user that wants to compare the effect of modifying various valve properties; for example, a user may want to find maximum stress in the piston for a given valve system as the thickness of the piston walls decrease. The user could easily write a Matlab script that sets the initial systems variables, then methodically alters the piston wall thickness, runs the simulation and Abaqus modelling function, and records the result. When enough data points are collected, the result could be plotted to show a maximum stress as a function of piston wall thickness. Examples of the user interface functions being used in this way are included in example two of this report.

Table 1 - User Interface Functions, Scripts and other files

Function or Script Name	Description
interface.m	Initiates the user interface. The user interface prompts the user for a task, then performs actions using the other functions and scripts to create the output requested by the user.
setVariables.m	Contains all variables that the simulation and FEA functions require to operate. A user can edit this file to describe their valve system.
runAbaqus.m	Starts model generation and FEA in Abaqus without the GUI. Requires three inputs: a name for the analysis, a time based pressure array (generated by the simulation), and a variable with the user input data required for Abaqus. Automatically creates a subfolder in the Matlab working directory and saves the job output to that folder.
runAbaqusGUI.m	Performs the same tasks and requires the same inputs as runAbaqus.m whilst also opening the model in the Abaqus GUI. Useful for automatically creating a model and immediately viewing FEA results, or modifying the generated design.
Other Files	
Dynamics.slx	The Simulink model that numerically calculates valve flow and dynamics properties. Requires all user input variables to exist in the Matlab workspace before running.
createPart.py	Python script that generates the piston model, applies a mesh and dynamic loads, and runs the FEA analysis job. Called by the functions runAbaqus.m and runAbaqusGUI.m.

B. Dynamics Modules

The dynamics modules of the model accept surface pressure values from the fluid mechanics module as well as geometric properties of the piston and projectile. The module outputs dynamics properties for each component using equations (1) specified in the physical principles section of this report. This module exists exclusively in the Simulink model, and requires the user input variables exist in the Matlab workspace to run. The piston and projectile both have separate modules as can be seen in Figure 3. Figure 4 shows the piston dynamics module with an example of the data it outputs.

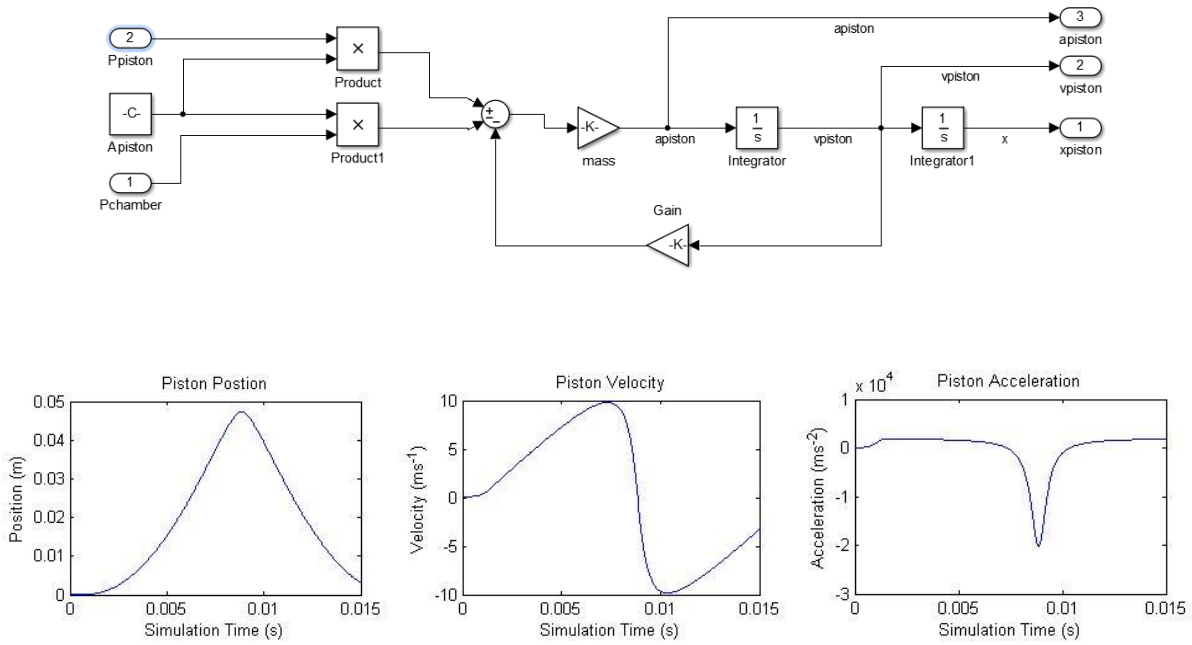


Figure 4 - Piston Dynamics Module and Example Output

C. Fluid Mechanics Module

Fluid mechanics calculations are handled by Simulink with embedded Matlab functions. The fluid mechanics module accepts piston and projectile position data from the dynamics module combined with fluid property variables to numerically calculate the fluid flow properties. Governing equations for this module can be found in equations (2) and (3). The fluid mechanic and dynamics modules communicate every iteration to accurately simulate the behavior of the valve. The fluid mechanics module is shown in Figure 5.

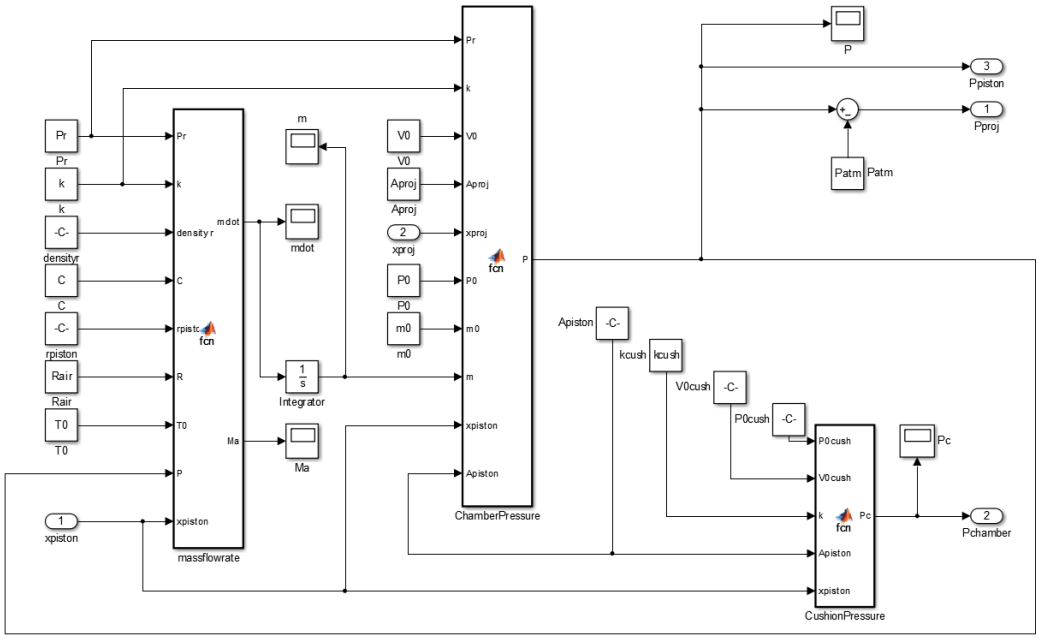


Figure 5 - Fluid Mechanics Simulink Model Module

D. Finite Element Analysis Module

The FEA module takes the user input variables and pressure data from the simulation, creates a three dimensional model and performs FEA in Abaqus utilising a Python script. The script automatically creates a part representing the piston, applies a mesh, applies a dynamic loading to each face of the piston and finally creates and runs a job. The script is run in the Matlab command line through the use of one of the two user input functions. This entire process is automated and requires no input from the user other than setting the initial variables in the Matlab workspace and calling one of the functions. Model generation and FEA can be done automatically with or without the Abaqus GUI depending on which function is called by the user.

The product of the script in the Abaqus GUI is demonstrated in example one.

E. Visualisation Module

The visualisation module of the Simulink model allows a user to view an animation of the piston and projectile moving relative to the chamber and cushion. An example of the module animation is shown in Figure 6. The red cylinder in the image represents the projectile, the blue cylinder the piston, and the black lines the chamber and cushion walls.

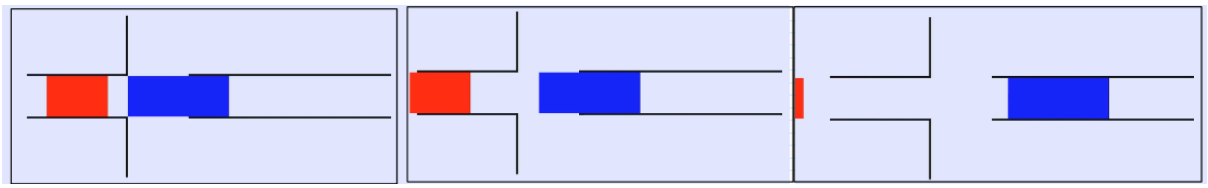


Figure 6 - Visualisation Module Animation Result

V. Examples of Model Usage

The proceeding sections provide examples of usage of the model in different scenarios. The aim of the examples is to demonstrate the robustness of the model, and how it may be used in a variety of ways in a future project involving the design of a quick opening valve for a single stage gas gun. Throughout these examples, reference is made to the user interface functions and scripts listed in Table 1.

A. Example One: Producing a single valve model with the user interface script

This example will show the process and result of using the user interface script *interface.m* to produce a valve model. The first step in this example is to modify the *setVariables.m* script to describe the properties of the valve to be modelled.

Figure 7 shows a sample of the variables that were edited in *setVariables.m*. A full copy of the script with variables is in Appendix C. The full list of variables used in this example is in Appendix A.

Once variables are set correctly, the *interface.m* script is run in the Matlab command window. The script will prompt the user to determine what kind of analysis is required. If an invalid command is entered, the script will simply prompt the user again to enter a valid command. Figure 8 shows the script requesting and accepting input from a user, executing a simulation, automatically generating a piston model in Abaqus, and finally performing FEA on the piston model in Abaqus. The user has the option to terminate the script at any stage, however in this example all elements of the script were utilised.

```
% Reservoir Properties

R=287.058;
Patm=101325;
Pr=500000+Patm;
T0r=20+273.15;
k=1.4;
```

Figure 7 - User variable input in *setVariables.m*.

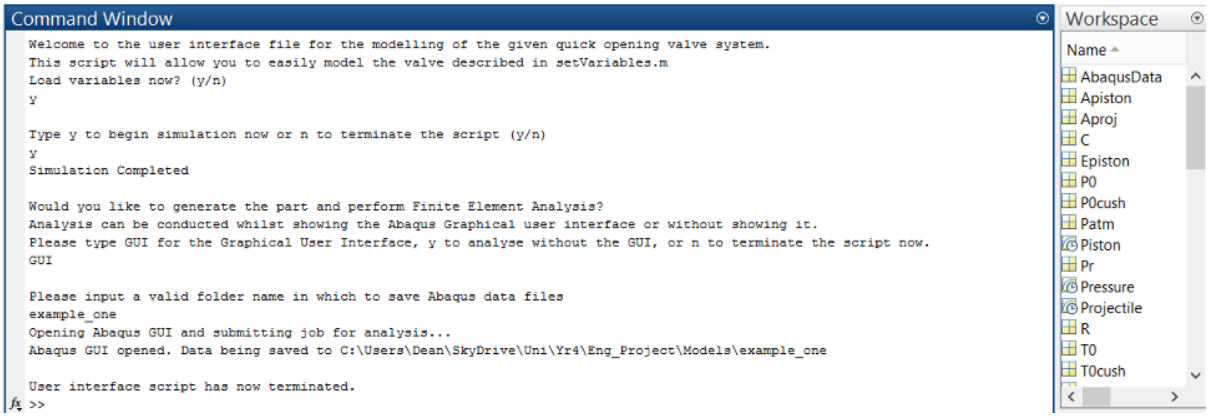


Figure 8 - User Interface script example

Because it was chosen to launch the Abaqus GUI, the GUI will be immediately launched and the results of the FEA will be displayed. Results are also saved in the folder defined by the user for later use. Figure 9 shows one frame of the Von Mises Stress results from the Abaqus GUI for this example.

A full set of dynamics and valve pressure data has now been stored in the Matlab workspace, and a model of the piston design with stress analysis data in Abaqus has been performed and saved. This data can be utilised as it is, or the model changed and the analysis run again.

This example demonstrates the ease with which a basic model can be generated, data analysed, and design modifications made.

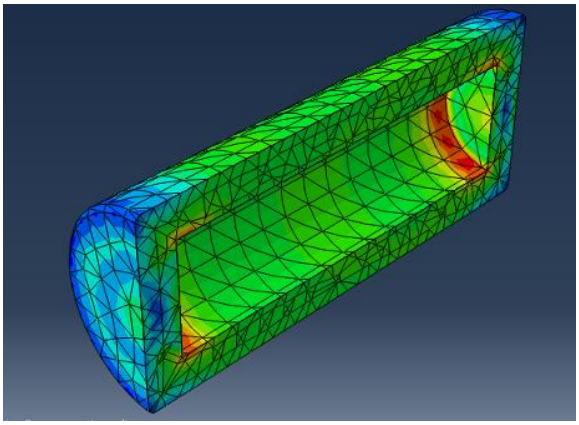


Figure 9 - Sample Von Mises stress analysis frame from Abaqus FEA for example one

B. Example Two: Automated comparison of dynamics properties for various reservoir pressures

The aim in this example is to simulate a variety of reservoir pressures and record the maximum acceleration experienced by the piston as well as the exit velocity of the projectile. To do this, a script is created that loads a set of variables describing the valve, then a simple for loop run that alters the reservoir pressure variable. The loop must then run the simulation and record the desired data, in this case the maximum absolute acceleration experienced by the piston and the maximum absolute velocity of the projectile. This data can then easily be plotted. An example of the script is shown in Figure 10. The results generated by the script are shown in Figure 11.

This example demonstrates how a user with a basic understanding of Matlab coding can generate large amounts of data very quickly. Note that for the sake of example only two variables were stored and plotted, but any number of dynamics or pressure variables could be calculated in the same fashion with minor modification to the script.

```
% Script for Example Two
clear
clc

setVariables
n=1;
for Pr=250000:10000:1000000
   sim('dynamics')
   resPressure(n)=Pr;
    maxAccelPiston(n)=max(abs(Piston.Data(:,3)));
    maxProjV(n)=max(abs(Projectile.Data(:,2)));
   n=n+1;
end

figure
plot(resPressure,maxProjV)
title('Projectile Exit Velocity for Varying
Reservoir Pressure')
xlabel('Reservoir Pressure')
ylabel('Projectile Exit Velocity')
```

Figure 10 - Example Two Script

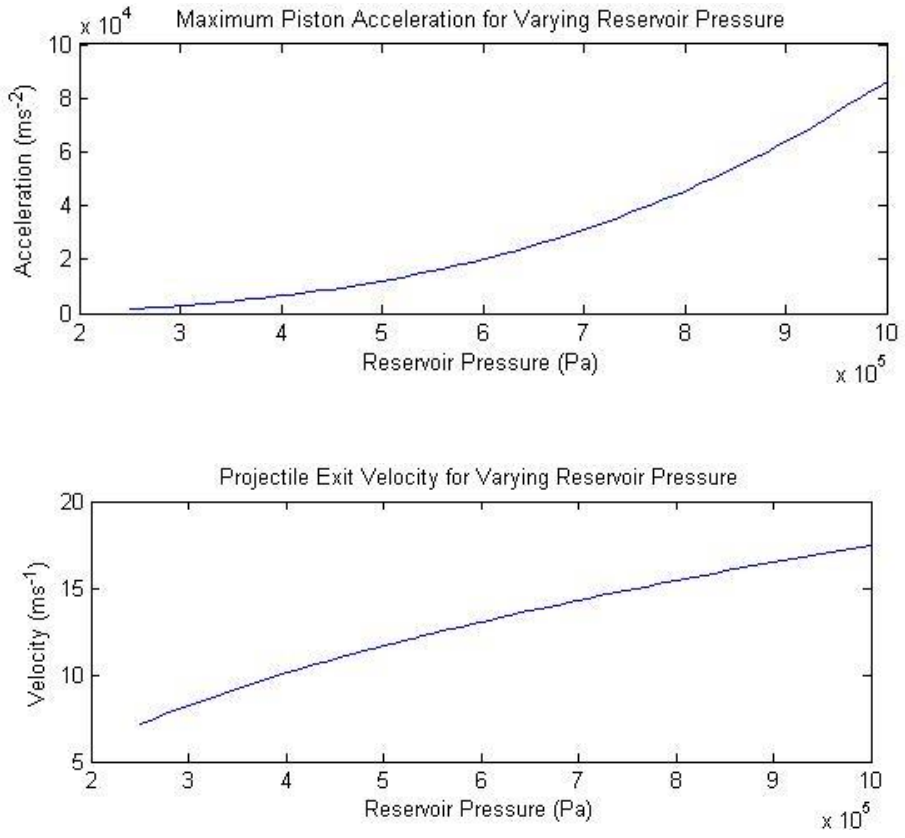


Figure 11 - Example Two Results

C. Example Three: Automated stress comparison for various working fluids

In this example, a Matlab script is created to compare the maximum stresses occurring in a given valve for various working fluids. The script also compares initial chamber flow properties for the fluids. Fluids investigated include air, nitrogen, helium, and hydrogen.

The script begins by loading the variables describing the valve. It then alters the fluid properties, runs a simulation of the valve dynamics, records the chamber pressure data, and finally utilises the *runAbaqus.m* function to automatically perform FEA on the piston. This process is repeated for the remaining working fluids. The script used to achieve this is shown in Figure 12.

The initial chamber pressure profiles for the various gases can be seen in Figure 13. This figure demonstrates how the property of a fluid affects the flow characteristic through the valve, and therefore chamber pressure profile during initial flow.

```
% Example Three
setVariables

% Run Sim for Air
R=287.058;
k=1.4;
sim('Dynamics')
pressAir=Pressure.Data(:,3);
runAbaqus('air',Pressure,AbaqusData)


% Run Sim for Nitrogen
R=297;
k=1.4;
sim('Dynamics')
pressNitrogen=Pressure.Data(:,3);
runAbaqus('nitrogen',Pressure,AbaqusData)

% Run Sim for Helium
R=2080;
k=1.667;
sim('Dynamics')
pressHelium=Pressure.Data(:,3);
runAbaqus('helium',Pressure,AbaqusData)

% Run Sim for hydrogen
R=4120;
k=1.4;
sim('Dynamics')
pressHydrogen=Pressure.Data(:,3);
runAbaqus('hydrogen',Pressure,AbaqusData)
```

Figure 12 - Example Three Script

Figure 14 shows the maximum Von Mises stress in the valve for air and helium. The result is as you would expect given the chamber pressure profiles shown in Figure 13. The higher flow of helium compared to air raises the chamber pressure faster, resulting in higher acceleration of the piston and as such higher internal stresses.

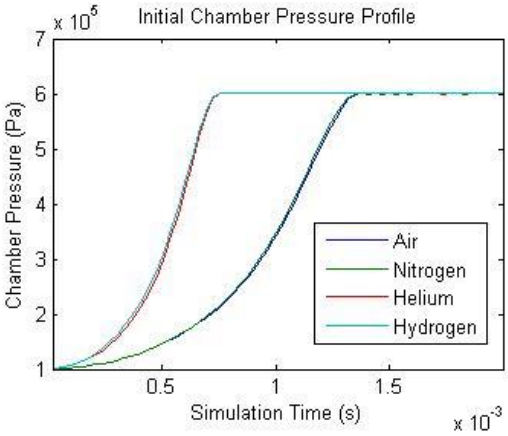
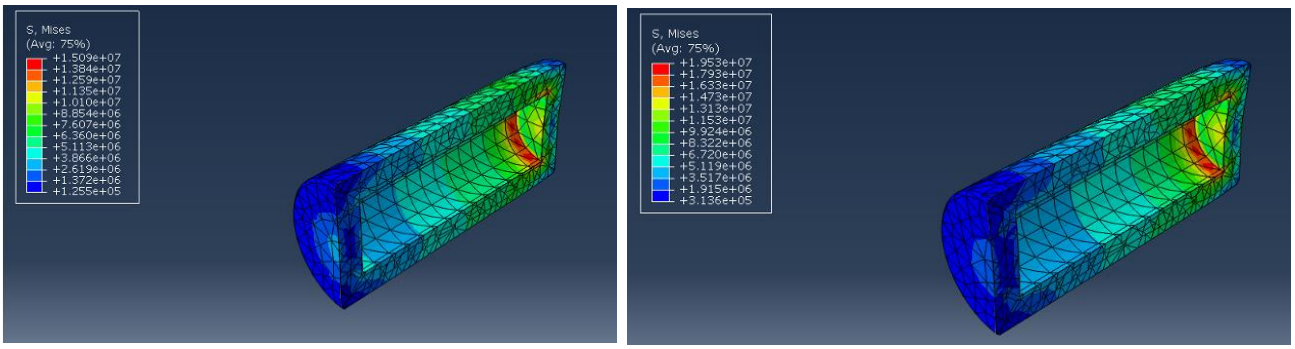


Figure 13 - Initial Chamber Pressure Profile for Various Gases



Air **Helium**

Figure 14 - Comparison of Stress Analysis Results for Air and Helium

This example has demonstrated the use of the *runAbaqus.m* function. It has shown the ability of the model to be used to quickly compare the effect of different valve parameters on the piston and projectile dynamics, as well as the piston internal stresses.

VI. Conclusions

The design of gas guns requires a complex analysis of the flow properties and dynamics of the valve system utilised. Fundamental physical principles governing valve dynamics have been identified and applied to create a set of software tools allowing a user to easily model an idealised quick opening valve within a single stage gas gun. This model allows a user to analyse multiple parameters within a valve design quickly with only a basic knowledge in Matlab encoding. Model functionality has been demonstrated through several examples.

VII. Recommendations

It is recommended that the model generated be passed to parties interested in the design of quick opening valves for single stage gas guns. The modular design of the model permits easy modification of any section of the model, allowing a user to improve a particular section of interest if required. As an example, a valve design may be particularly dependent on the fluid flow prior to pressure equalization between the chamber and reservoir. In this case, the designer may wish to modify the fluid module to a more in depth computational fluid dynamics module. This would be achievable whilst leaving the remainder of the model intact.

Further work could include comparison of the model to real systems to ascertain the effect of idealization of the fluid mechanics and dynamics within the model. Research pertaining to the model could include experimental determination of valve coefficients for isentropic flow compensation for a range of valve flow profiles.

References

Cai, R., 1998. *Some Explicit Analytical Solutions of Unsteady Compressible Flow,* Beijing: Institute of Engineering Thermophysics.

Cengel, Y. A., Cimbala, J. M. & Turner, R. H., 2012. *Fundamentals of Thermal-Fluid Sciences.* Fourth ed. Singapore: Mc Graw Hill.

Engineering Design, 2011. *Control Valves.* [Online]
Available at: http://www.enggcyclopedia.com/2011/05/control-valves/

Headley, M. C., 2003. Guidelines for Selecting the Proper Valve Characteristic. *Valve Magazine.*

Hibbeler, R. C., 2010. *Engineering Mechanics Dynamics.* Twelfth in SI Units ed. Singapore: Prentice Hall.

Honeywell, 1976. *Pneumatic Temperature Controls Theory Manual.* s.l.:Honeywell.

Jad, A., 2014. *Plug Valve: An Isolation Valve.* [Online]
Available at: http://www.piping-engineering.com/plug-valves-isolation-valve.html

Janson, C., 2008. *Selection, Sizing, and Operation of Control Valves for Gases and Liquids,* Marshalltown, Iowa USA: Fisher Controls International Inc.

Munson, B. R., Young, D. F., Okiishi, T. H. & Huebsch, W. W., 2010. *Fundamentals of Fluid Mechanics.* Sixth ed. New Jersey: John Wiley & Sons.

Naval Education and Training Command, 1990. *Fluid Power,* United States: NAVEDTRA.

Samson AG, 2012. *Application Notes - Kv coeffecient, valve sizing.* [Online]
Available at: http://www.samson.de/pdf_en/t00050en.pdf

Seidl, M., Hughes, K. & De Vuyst, T., 2013. *Modelling internal gas flows in a single stage gas gun using Eularian/Lagrangian coupling in LS-DYNA,* Bedforshire: Department of Applied Mechanics and Astronautics,Cranfield University.

Wilson, C. E. & Peter, J. S., 2006. *Kinematics and Dynamics of Machinery.* Third ed. Singapore: Pearson Prentice Hall.