Novel Approaches to Fast Filling of Hydrogen Cylinders

Pau Miquel Mir
28023668

Supervisor: Dr. Edward Richardson

Word count: 466

May 2018

This report is submitted in partial fulfillment of the requirements for the MEng Mechanical Engineering, Faculty of Engineering and the Environment, University of Southampton.

Declaration

I, Pau Miquel Mir, declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research. I confirm that:

1. This work was done wholly or mainly while in candidature for a degree at this University;
2. Where any part of this thesis has previously been submitted for any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. None of this work has been published before submission.

Acknowledgements

I want to thank my advisor for his time and dedication.
I also want to thank my parents, for their constant support and for proofreading the paper.

Abstract

This is the abstract.

Contents

Acronyms

ABS Anti-lock Braking System

SVM Support Vector Machine

1 Introduction

Hydrogen is a very promising alternative fuel for the future, mainly due to the absence of greenhouse emissions when burning it. There are however several concerns with hydrogen fuels, such as: risk of carried compressed fuel, obtaining hydrogen itself, etc. However, one of the most important aspects is the convenience and user experience. Indeed, even if a new technology is scientifically better, consumers will prefer an inferior technology that is more convenient to them, especially if the improved technology doesn?t directly affect them (as reduced emissions don?t). For this reason, it is essential for the success of hydrogen as a fuel for its use to be as ? if not more ? convenient than traditional fuel. One of the main aspect?s that currently lags behind traditional fuel is the refueling experience. Given that refueling hydrogen involves its compression, there is a significant rise in temperature, which must be kept below certain standards (358K as per SAE J2601). This in turn leads to long refueling times, potentially lasting more than five minutes, which is cumbersome for users. Therefore, it is of prime importance to research and develop systems that enable the faster refueling of hydrogen cylinders. To this end, this project will build upon a model of filling a hydrogen cylinder which has already been developed by members of the department.

One of the current solutions to improve fill times involves cooling the hydrogen before filling the cylinder as to keep it below he maximum temperature. However, this is quite expensive, both in energy terms and in economic terms. Consequently, the aim of this project is to continue exploring several of the options available to reduce fill times and simultaneously reduce the energy consumption of the process, thus improving both convenience for users and energy efficiency of the fueling stations. Indeed, by building upon the existing cylinder model several options shall be considered, namely: refrigeration, flow regulation, heat sink usage, active cooling, heat pipe usage, phase change materials, etc. From here, several options are open to further deepen or potentially broaden the investigation. An attempt to further simplify the model can be made, perhaps even reducing it so a simple algebraic relationship. Also, the model would benefit from FEA validation to aid our understanding of the heat transfer in the structure. This would go hand in hand with analyzing the temperature of the structure, and see how close this matches the gas temperature. Indeed, if the structure is at a much lower temperature than the gas, the case can be made that the current regulations are slightly erroneous, as they are meant to protect the materials of the structure, but instead regulate the gas temperature.

2 Conclusion

This is the conclusion. This is shit. This is crap. Fuck fuck this is boring. Three more words.

Appendices

A Prova

This is a new Appendix.

$$3+3=6 \tag{A.1}$$

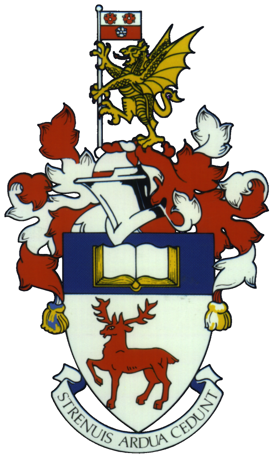

FIGURE A.1: Prova

1. Title page pretty

2. Word count glossaries

A.1 Proves

First use: Support Vector Machine (SVM). Second use: SVM.

First use: Anti-lock Braking System (ABS). Second use: ABS.

Prova [1]. Also, here is a trial on Fig. A.2

This is a piece of shit. Fuck Prova merda I just keep on typing and want to see what happens, there is simply a bit of a lag and that it and that's it, shit this isn't working. It is, it is simply very very slow. I want to see if I can keep on typing and see what happens, it is quite nice indeed. How many words can I get without it working this is pretty cool and now I want to in insert a reference to Fig A.2 This is interesting. What if I want to [1], and also [2], finally **Nelder1965**

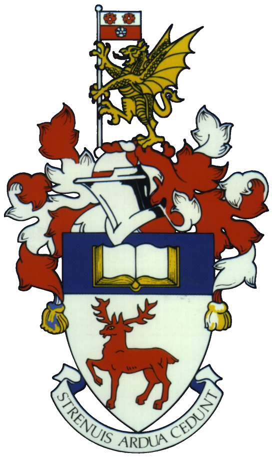

FIGURE A.2: This caption is really really long and uses up more than one line to test the caption package to see what the fuck happens. This is a test. Caca.

In Fig. A.2 we can see a shitty coat of arms. I am going to add eight more words. afjhffa

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.2}$$

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

B Eurobot Code

```
%Author: Vishagen Ramasamy
%Date: -

% Copyright University of Southampton 2017.
% No warranty either expressed or implied is given to the results produced
% by this software.  Neither the University, students or its employees
%accept any responsibility for use of or reliance on results produced by
%this software.

%% script that computes the filling of the tank(s)
%% Importing the pressure and temperature profile at the entrance of the delivery pipe from Dicken a
%  and calulating the stagnation enthalpy and entropy

for j = 1:maxt
    time(j+1) = (j)*dt;                                                        % Time as filling proce
    for i = 1:tank_number
        % Select inlet pressure / temperature profile based on user input
        if blnUseStandardData == 1
```

```
            % Constant inlet pressure profile
            P_inlet(i,j+1) = ConstPressKPa;
        else
            % Read inlet pressure profile from specified file
            P_inlet(i, j+1) = interp1(InletPressureData(:,1),InletPressureData(:,3), dt*j);
        end

        if blnUseStandardData == 1
            %Temp_inlet(i,j+1) = interp1(time_in, temp_in, dt*j);   % Temperature profile after the first 1.35 s
            Temp_inlet(i, j+1) = ConstTempK;
        else
            % Read inlet temperature profile from specified file
            Temp_inlet(i, j+1) = interp1(InletTempData(:,1), InletTempData(:,3), dt*j);
        end

        h_inlet(i,j+1) = refpropm('H','T',Temp_inlet(i,j+1),'P',P_inlet(i,j+1),Fluid{i}, refpropdir);       % Re
        entropy_inlet(i,j+1) = refpropm('S','T',Temp_inlet(i,j+1),'P',P_inlet(i,j+1),Fluid{i}, refpropdir); % Re

        %% Determine which pressure to use at the exit of the delivery pipe which is dependent upon the number o

        if l_d(i) > 3 && blnOneZone{i} == 0          % The length-to-diameter ratio of the tank(s) determines th
            Pressure_exit = P_gas_zone1{i}(j);   % The pressure at the exit of the delivery pipe is equal to the
        else
            Pressure_exit = P_gas(i,j);          % The pressure at the exit of the delivery pipe is equal to the
        end

        if (P_inlet(i,j+1) > Pressure_exit)  % condition for filling of tank(s)
            sound_exit(i,j+1) = refpropm('A','P',Pressure_exit,'S',entropy_inlet(i,j+1),Fluid{i}, refpropdir);
            h_static_exit(i,j+1) = refpropm('H','P',Pressure_exit,'S',entropy_inlet(i,j+1),Fluid{i}, refpropdir)
            visc_exit(i,j+1) = refpropm('V','P',Pressure_exit,'S',entropy_inlet(i,j+1),Fluid{i}, refpropdir);
            mach_exit(i,j+1) = sqrt(2*(h_inlet(i,j+1)-h_static_exit(i,j+1))/sound_exit(i,j+1)^2);   % Calculates

            %% If Mach number is greater than one, the exit pressure is greater than the pressure within the tan
            %  is incremetally increased and iterated in a while loop  until mach number is equal to one

            Inlet_entropy = entropy_inlet(i,j+1);
            Inlet_stagnation_enthalpy = h_inlet(i,j+1);
            P_guess =  Pressure_exit;

            if mach_exit(i,j+1) > 1
                Pressure_exit = find_exit_pressure(Inlet_stagnation_enthalpy,Inlet_entropy,Fluid{i},P_guess, ref
                sound_exit(i,j+1) = refpropm('A','P',Pressure_exit,'S',Inlet_entropy,Fluid{i}, refpropdir);
                h_static_exit(i,j+1) = refpropm('H','P',Pressure_exit,'S',Inlet_entropy,Fluid{i}, refpropdir);
                visc_exit(i,j+1) = refpropm('V','P',Pressure_exit,'S',Inlet_entropy,Fluid{i}, refpropdir);
                mach_exit(i,j+1) = sqrt(2*(h_inlet(i,j+1)-h_static_exit(i,j+1))/sound_exit(i,j+1)^2);       % Co
            end

            %% Calculation of the mass flow rate into the tank(s)

            rho_exit(i,j+1) = refpropm('D','P',Pressure_exit,'S',Inlet_entropy,Fluid{i}, refpropdir);
            vel_exit(i,j+1) = mach_exit(i,j+1)*sound_exit(i,j+1);                                        % Calcul
            Re_exit_isentropic(i,j+1) = rho_exit(i,j+1) *d_inlet*vel_exit(i,j+1)/visc_exit(i,j+1);       % Calcul
            cd(i,j+1) =  I + J/(Re_exit_isentropic(i,j+1)^(0.25));                                       % Calcul
            mfr(i,j+1) = cd(i,j+1)*rho_exit(i,j+1)*vel_exit(i,j+1)*A_inlet;                              % Calcul
            Re_entrance_actual(i,j+1)= 4*mfr(i,j+1)/(pi*d_inlet*visc_exit(i,j+1));                       % Calcul
            dM_inlet = mfr(i,j+1)*dt;                                                                    % Amount
        end

        %% Heat transfer calculations & caluclations of the thermodynamic properties of the gas in the tank(s) w
        if l_d(i) <= 3 | blnOneZone{i} == 1

            Nus(i,j+1) =   a_1*Re_entrance_actual(i,j+1)^(b_1);                     % Nusselt number and Reynolds
            k_gas(i,j+1) = refpropm('L','T',Temp_gas(i,j),'P',P_gas(i,j),Fluid{i}, refpropdir);  % Thermal condu
            heat_coef_forced(i,j+1) = Nus(i,j+1)*k_gas(i,j+1)/d_tank(i);            % Calculation of the heat tran

            if Inner_wall_boundary(i) == 1
```

```
                Qsurf(i,j+1) = -dt*surf_area(i)*heat_coef_forced(i,j+1)*(Temp_gas(i,j)-Inner_temp_wa
            else
                Qsurf(i,j+1) = -dt*surf_area(i)*heat_coef_forced(i,j+1)*(Temp_gas(i,j)-Temp_wall{i}(
                Temp_wall{i}(1,j+1) = Temp_wall{i}(1,j)+ CFL_liner(i)*(Temp_wall{i}(2,j)- Temp_wall{

                if Outer_wall_boundary(i)==1
                    Temp_wall{i}(number_of_gridpoints(i),j+1) = Outer_temp_wall_isothermal(i);
                else
                    Temp_wall{i}(number_of_gridpoints(i),j+1) = Temp_wall{i}(number_of_gridpoints(i)
                end
                % Computation of the temperature of the struture of the tank(s)
                for k=2:number_of_gridpoints(i)-1
                    if (k>=2)&&(k<=int_pt_liner_laminate(i)-1)
                        Temp_wall{i}(k,j+1) = Temp_wall{i}(k,j)+CFL_liner(i)*(Temp_wall{i}(k+1,j)-2*
                    elseif (k>=int_pt_liner_laminate(i)+1)&&(k<=number_of_gridpoints(i)-1)
                        Temp_wall{i}(k,j+1) = Temp_wall{i}(k,j)+CFL_laminate(i)*(Temp_wall{i}(k+1,j)
                    else
                        Temp_wall{i}(k,j+1) = (cond_laminate(i)*(Temp_wall{i}(k+1,j)+CFL_laminate(i)
                    end
                end
            end
            m_gas(i,j+1) = m_gas(i,j)+ dM_inlet;                                             % Mas
            Ugas(i,j+1)=Ugas(i,j)+Qsurf(i,j+1)+h_inlet(i,j+1)*dM_inlet;                      % Int
            u_gas(i,j+1)= Ugas(i,j+1)/m_gas(i,j+1);                                          % Spe
            rho_gas(i,j+1)=m_gas(i,j+1)/vol_tank(i);                                         % Den
            P_gas(i,j+1) = refpropm('P','D',rho_gas(i,j+1),'U',u_gas(i,j+1),Fluid{i}, refpropdir);
            Temp_gas(i,j+1) = refpropm('T','D',rho_gas(i,j+1),'U',u_gas(i,j+1),Fluid{i}, refpropdir)

        else
            Re_compression(i,j+1) = Re_entrance_actual(i,j+1)*(d_inlet/d_tank(i));          % Rey

            Nus_zone1{i}(j+1) =  a_1*Re_entrance_actual(i,j+1)^(b_1);                        % Nus
            Nus_zone2{i}(j+1) =  c_1*Re_compression(i,j+1)^(d_1);                            % Nus

            k_gas_zone1{i}(j+1) =  refpropm('L','T',Temp_gas_zone1{i}(j),'P',P_gas_zone1{i}(j),Fluid
            k_gas_zone2{i}(j+1) =  refpropm('L','T',Temp_gas_zone2{i}(j),'P',P_gas_zone2{i}(j),Fluid

            heat_coef_forced_zone1{i}(j+1) =  Nus_zone1{i}(j+1)*k_gas_zone1{i}(j+1)/d_tank(i);
            heat_coef_forced_zone2{i}(j+1) =  Nus_zone2{i}(j+1)*k_gas_zone2{i}(j+1)/l_zone2_total(i)

            % Step 1. Equating the respecting values of zone 1 and zone 2 to
            % matrices with 'similar' names

            Vgas(1)=volume_zone1(i);
            Vgas(2)=volume_zone2(i);
            Mgas(1)=m_gas_zone1{i}(j);
            Mgas(2)=m_gas_zone2{i}(j);
            Ugas_twozone(1)=u_gas_zone1{i}(j)*Mgas(1);
            Ugas_twozone(2)=u_gas_zone2{i}(j)*Mgas(2);
            Asurf(1)=surface_area_zone1(i);
            Asurf(2)=surface_area_zone2(i);
            Tgas(1)=Temp_gas_zone1{i}(j);
            Tgas(2)=Temp_gas_zone2{i}(j);
            if Inner_wall_boundary(i) == 1
                Twall(1,1)=Inner_temp_wall_isothermal(i);
                Twall(2,1)=Inner_temp_wall_isothermal(i);
            else
                Twall(1,1)=Temp_wall_zone1{i}(1,j);
                Twall(2,1)=Temp_wall_zone2{i}(1,j);
            end
            % Step 2: apply the change of internal energy due to heat transfer and mass
            % input through nozzle:

            dM_inlet = mfr(i,j+1)*dt;

            Qsurf(1)=- dt*Asurf(1)*(heat_coef_forced_zone1{i}(j+1))*(Tgas(1)-Twall(1,1));
```

```
            Qsurf(2)=- dt*Asurf(2)*(heat_coef_forced_zone2{i}(j+1))*(Tgas(2)-Twall(2,1));

            Ugas_twozone(1)=Ugas_twozone(1)+Qsurf(1)+h_inlet(i,j+1)*dM_inlet;
            Ugas_twozone(2)=Ugas_twozone(2)+Qsurf(2);

            Mgas(1)=Mgas(1)+dM_inlet;
            Mgas(2)=Mgas(2);

            for m=1:2
                hgas(m)=refpropm('H','D',Mgas(m)/Vgas(m),'U',Ugas_twozone(m)/Mgas(m),Fluid{i}, refpropdir);
            end


            % Step 2: Find the amount of mass that needs to be transferred from zone 1
            % to zone 2 to equalise their pressure
            % (we assume forward Euler integration, therefore we use the specific
            % enthalpies h from the start of the timestep. We provide enthalpies for
            % both zones in case the flow gets reversed).

            dM_guess = dM_inlet*(volume_zone2(i)/(volume_zone1(i)+volume_zone2(i))) ;
            dM_12 = find_dM_12(hgas,Vgas,Mgas,Ugas_twozone,Fluid{i},dM_guess, refpropdir);

            % Step 3: Apply this change to the mass and update all properties:

            m_gas_zone1{i}(j+1)=Mgas(1)-dM_12;
            m_gas_zone2{i}(j+1)=Mgas(2)+dM_12;

            u_gas_zone1{i}(j+1)=(Ugas_twozone(1)-max(0,dM_12)*hgas(1)-min(0,dM_12)*hgas(2))/m_gas_zone1{i}(j+1)
            u_gas_zone2{i}(j+1)=(Ugas_twozone(2)+max(0,dM_12)*hgas(1)+min(0,dM_12)*hgas(2))/m_gas_zone2{i}(j+1)

            rho_gas_zone1{i}(j+1)=m_gas_zone1{i}(j+1)/volume_zone1(i);
            rho_gas_zone2{i}(j+1)=m_gas_zone2{i}(j+1)/volume_zone2(i);

            Temp_gas_zone1{i}(j+1)=refpropm('T','D',rho_gas_zone1{i}(j+1),'U', u_gas_zone1{i}(j+1),Fluid{i}, ref
            Temp_gas_zone2{i}(j+1)=refpropm('T','D',rho_gas_zone2{i}(j+1),'U', u_gas_zone2{i}(j+1),Fluid{i}, ref

            P_gas_zone1{i}(j+1)=refpropm('P','D',rho_gas_zone1{i}(j+1),'U', u_gas_zone1{i}(j+1),Fluid{i}, refpro
            P_gas_zone2{i}(j+1)=refpropm('P','D',rho_gas_zone2{i}(j+1),'U', u_gas_zone2{i}(j+1),Fluid{i}, refpro

            m_gas(i,j+1) = m_gas_zone1{i}(j+1) + m_gas_zone2{i}(j+1);
            rho_gas(i,j+1) = m_gas(i,j+1)/vol_tank(i);
            u_gas(i,j+1)= (u_gas_zone1{i}(j+1)*m_gas_zone1{i}(j+1)+ u_gas_zone2{i}(j+1)*m_gas_zone2{i}(j+1))/m_g
            Ugas(i,j+1) = u_gas(i,j+1)*m_gas(i,j+1);
            Temp_gas(i,j+1)  = refpropm('T','D',rho_gas(i,j+1),'U',u_gas(i,j+1),Fluid{i}, refpropdir);
            P_gas(i,j+1)  = refpropm('P','D',rho_gas(i,j+1),'U',u_gas(i,j+1),Fluid{i}, refpropdir);

            if Inner_wall_boundary(i) == 1
                Qsurf_zone1{i}(j+1) = -dt*surface_area_zone1(i)* heat_coef_forced_zone1{i}(j+1)*(Temp_gas_zone1
                Qsurf_zone2{i}(j+1) = -dt*surface_area_zone2(i)* heat_coef_forced_zone2{i}(j+1)*(Temp_gas_zone2
            else
                Qsurf_zone1{i}(j+1) = -dt*surface_area_zone1(i)* heat_coef_forced_zone1{i}(j+1)*(Temp_gas_zone1
                Qsurf_zone2{i}(j+1) = -dt*surface_area_zone2(i)* heat_coef_forced_zone2{i}(j+1)*(Temp_gas_zone2

                Temp_wall_zone1{i}(1,j+1) = Temp_wall_zone1{i}(1,j)+ CFL_liner(i)*(Temp_wall_zone1{i}(2,j)- Temp
                Temp_wall_zone2{i}(1,j+1) = Temp_wall_zone2{i}(1,j)+ CFL_liner(i)*(Temp_wall_zone2{i}(2,j)- Temp

                if Outer_wall_boundary(i)==1
                    Temp_wall_zone1{i}(number_of_gridpoints(i),j+1) = Outer_temp_wall_isothermal(i);
                    Temp_wall_zone2{i}(number_of_gridpoints(i),j+1) = Outer_temp_wall_isothermal(i);
                else
                    Temp_wall_zone1{i}(number_of_gridpoints(i),j+1) = Temp_wall_zone1{i}(number_of_gridpoints(i)
                    Temp_wall_zone2{i}(number_of_gridpoints(i),j+1) = Temp_wall_zone2{i}(number_of_gridpoints(i)
                end
                % Computation of the temperature of the struture of the tank(s)
                for k=2:number_of_gridpoints(i)-1
                    if (k>=2)&&(k<=int_pt_liner_laminate(i)-1)
```

```
                        Temp_wall_zone1{i}(k,j+1) = Temp_wall_zone1{i}(k,j)+CFL_liner(i)*(Temp_wall_
                        Temp_wall_zone2{i}(k,j+1) = Temp_wall_zone2{i}(k,j)+CFL_liner(i)*(Temp_wall_
                    elseif (k>=int_pt_liner_laminate(i)+1)&&(k<=number_of_gridpoints(i)-1)
                        Temp_wall_zone1{i}(k,j+1) = Temp_wall_zone1{i}(k,j)+CFL_laminate(i)*(Temp_wa
                        Temp_wall_zone2{i}(k,j+1) = Temp_wall_zone2{i}(k,j)+CFL_laminate(i)*(Temp_wa
                    else
                        Temp_wall_zone1{i}(k,j+1) = (cond_laminate(i)*(Temp_wall_zone1{i}(k+1,j)+CFL
                        Temp_wall_zone2{i}(k,j+1) = (cond_laminate(i)*(Temp_wall_zone2{i}(k+1,j)+CFL
                    end
                end
            end

        end

    end
end
```

References

[1] Y.-L. Liu, Y.-Z. Zhao, L. Zhao, X. Li, H.-g. Chen, L.-F. Zhang, H. Zhao, R.-H. Sheng, T. Xie, D.-H. Hu, and J.-Y. Zheng, "Experimental studies on temperature rise within a hydrogen cylinder during refueling", *International journal of hydrogen energy*, vol. 35, no. 7, pp. 2627–2632, Apr. 2010, ISSN: 03603199. DOI: 10.1016/j.ijhydene.2009.04.042. [Online]. Available: https://www.engineeringvillage.com/share/document.url?mid=cpx%7B%5C_%7D6e3d601283fcb129fM7f742061377553%7B%5C&%7Ddatabase=cpx.

[2] "Thermal characteristics during hydrogen fueling process of type IV cylinder", *International journal of hydrogen energy*, vol. 35, no. 13, pp. 6830–6835, Jul. 2010, ISSN: 0360-3199. DOI: 10.1016/J.IJHYDENE.2010.03.130. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0360319910006476?%7B%5C_%7Drdoc=1%7B%5C&%7D%7B%5C_%7Dfmt=high%7B%5C&%7D%7B%5C_%7Dorigin=gateway%7B%5C&%7D%7B%5C_%7Ddocanchor=%7B%5C&%7Dmd5=b8429449ccfc9c30159a5f9aeaa92ffb%7B%5C&%7Dccp=y.