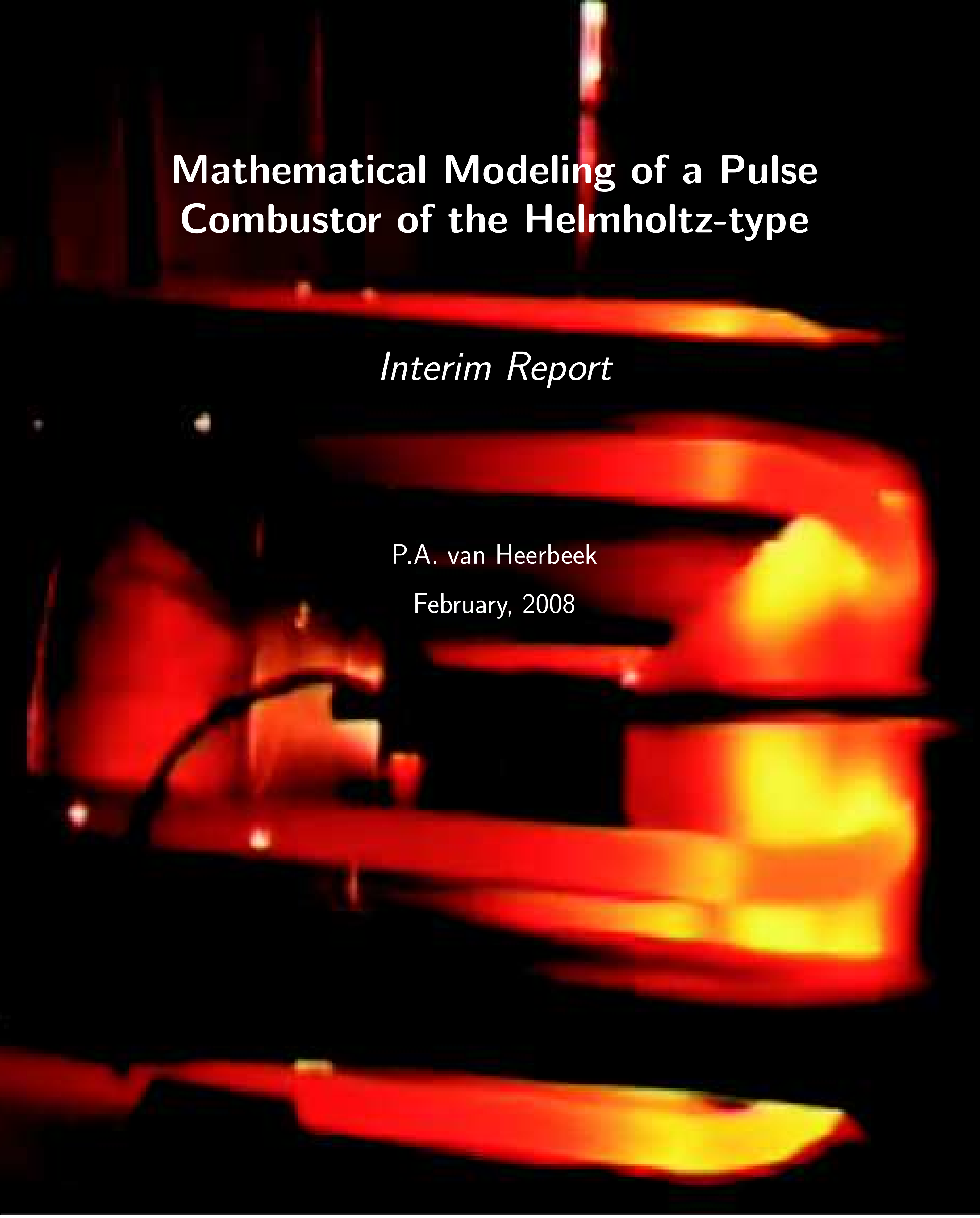


Mathematical Modeling of a Pulse Combustor of the Helmholtz-type

Interim Report

P.A. van Heerbeek

February, 2008

DLF Sustainable

Sustainable Technology & Environmental Solutions

TUDelft

Delft University of Technology

Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Mathematical modeling of a pulse combustor of the Helmholtz-type

Interim Report

P.A. van Heerbeek

Delft, The Netherlands
February 2008

A joint research project
DLF Sustainable/Delft University of Technology

DLF Sustainable
Sustainable Technology & Environmental Solutions

TU Delft
Delft University of Technology

Members of the Master's Thesis Committee:

Responsible professor:

Prof.dr.ir. C. Vuik (Delft University of Technology)

Daily supervisor:

Dr.ir. M.B. van Gijzen (Delft University of Technology)

Other thesis committee members:

Ir. M.R. de la Fonteijne (DLF Sustainable)
Ir. H.F.M. Corstens (Delft University of Technology)

Contact information

Delft University of Technology

P.A. van Heerbeek

Delft University of Technology
Delft Institute of Applied Mathematics
Mekelweg 4 (Room HB07.030)
2628 CD Delft
The Netherlands

E-mail: `P.A.vanHeerbeek@student.tudelft.nl`

DLF Sustainable
Sustainable Technology & Environmental Solutions

Ir. M.R. de la Fonteijne

DLF Sustainable
P.O. Box 1077
2600 BB Delft
The Netherlands

Mobile: +31 621885101
E-mail: `marcel@dlfsustainable.nl`
Web-site: `www.dlfsustainable.nl`

Mathematical modeling of a pulse combustor of the Helmholtz-type

Interim Report

Preface

This interim report is part of my Master's thesis project for the study "Applied Mathematics" at the Delft University of Technology (TU Delft). The project is conducted in collaboration with DLF Sustainable and aims at developing mathematical tools to aid in the design of pulse combustors and to gain insight into the physical processes that make them work (or not). An ultimate goal is the development of a mathematical model for an advanced industrial pulse combustor developed by DLF Sustainable (see Figure 1a).

With this report an orientation phase comes to an end. It presents some results of a literature study on the subject of pulse combustion and formulates research questions raised during the literature study and possible directions which the ensuing research phase of the project may take.

At the start of the current research project, DLF Sustainable was already involved in a joint research project on pulse combustion with the Eindhoven University of Technology (TU Eindhoven). A small pulse combustor has been made available for experimenting (see Figure 1b). The pulse combustor is much simpler in design than the industrial pulse combustors produced by DLF Sustainable, but the working principles are basically the same. Research at the TU Eindhoven is mainly progressed via Bachelor's thesis projects, which are concentrated on zero-dimensional models of pulse combustors from literature.

For the current project, the collaboration is extended to the TU Delft. Research at the TU Eindhoven and at the TU Delft will be focusing on different aspects of pulse combustion. Work at the TU Delft will be concentrating on a

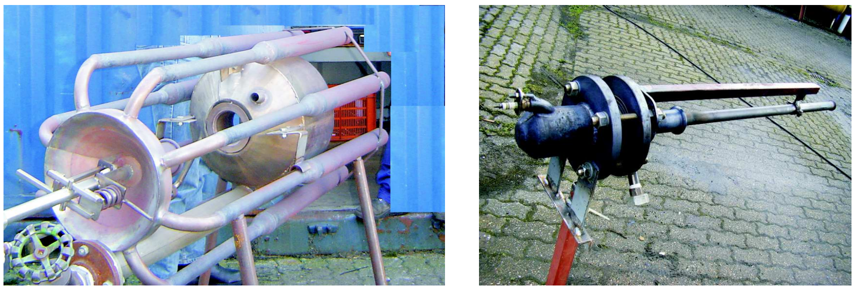

(a) An advanced pulse combustor (b) Small pulse combustor at TUE

Figure 1: Two pulse combustors: (a) an advanced pulse combustor developed by DLF Sustainable, (b) a small pulse combustor made available for experimenting at the TU Eindhoven (TUE); photo (b) is from Tiemessen and Bastiaans [32].

mathematical description of the acoustics of a pulse combustor, and how it is affected by heat release. Work at the TU Eindhoven will be concentrating on combustion processes in the combustion chamber. Since a typical pulse combustor has a relatively compact combustion chamber compared with the length of the tailpipe and associated wavelengths, it may be expected that the spatial distribution of heat release by combustion will not be of great influence on the system's acoustics. Vice versa, since the acoustic fluctuations will be almost uniform in a compact combustion chamber, it may be expected that the acoustics will not be of great influence on the spatial characteristics of the combustion process. Thus, for a large part the research on acoustics and combustion processes in a pulse combustor may be done independently, though the modeling results need to be coupled in the end. Experimental research with the pulse combustor at the TU Eindhoven can be useful for validation of a mathematical model.

I have taken note of two Bachelor's theses on pulse combustion [32, 34], produced at the TU Eindhoven. They will be mentioned at the appropriate places in this report.

Pieter van Heerbeek,
February 2008, Delft.

DLF Sustainable is a specialist in the field of thermodynamics, with a focus on sustainable energy generation and energy saving in an industrial environment. Keywords are: biomass gasification, pyrolysis, CHP, solar power, PV, cogen.

Because pulse combustion can have a positive impact on reducing environmental emission, DLF Sustainable asked Pieter van Heerbeek to investigate pulse combustion from a theoretical point of view. In this preliminary study Pieter van Heerbeek examined thoroughly several models from literature, in order to be capable of defining goals for the second part of his Master's thesis. So far, this resulted in several corrections of models shown in literature, the results of which are presented in a poster session on the Combura 2007 symposium at NBC in Nieuwegein (Holland) on the 10th of October 2007.

I am very pleased to see the enthusiasm of Pieter doing this study and I appreciate his sound working attitude. We are looking forward to continue this research project with Pieter.

Marcel de la Fonteijne,
DLF Sustainable.

Contents

Chapter 1

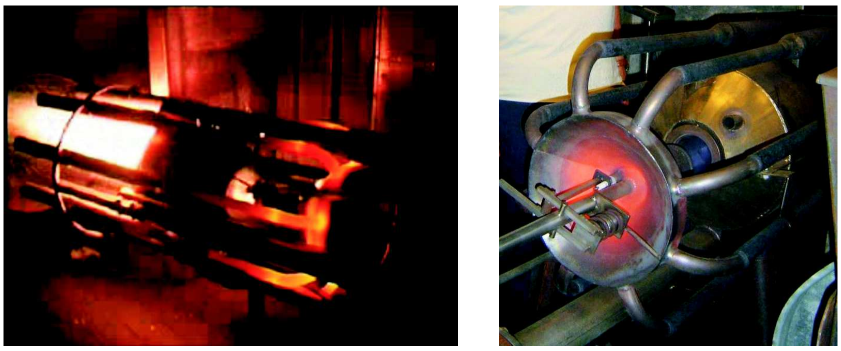
Introduction

1.1 Background and motivation

The company DLF Sustainable in Delft is a specialist in the field of thermodynamic applications. Examples of industrial applications are: gas fired installations (for direct or indirect heat generation, central heating boilers), dryers (for feed stock, industrial sewage/sludge), incinerators (for hazardous waste, thermal oxidation of gases/fluids/solids) and gasifiers (for biomass gasification). This list is by no means exhaustive.

During years of experience in the field and doing research, DLF Sustainable has developed an advanced pulse combustor for industrial applications such as mentioned above (see Figure 1.1). Compared to conventional (steady flow) combustion systems, this type of combustor can achieve production increases as high as 40%, and fuel savings of 10–50%, depending on the application. Also, it produces very low emissions of harmful substances. These advantages are typical of pulse combustors and can be related to the generation of intense oscillatory fluid motions (sound waves) in the combustion system, which is a fundamental part of the operating conditions. The high-intensity sound waves cause enhanced heat and mass transfer, higher combustion intensities, lower emissions of nitrogen oxides (NO_x) and higher thermal efficiencies, all by a considerable factor of magnitude compared to conventional combustion systems [9,10,14,16,23]. Also, usually pulse combustors provide their own means for pumping fuel and air, obviating the need for extra devices such as fans or a chimney. This may lead to more compact designs. The mentioned advantages make pulse combustion a practically, economically and environmentally attractive technique for burning and heating applications.

There are disadvantages to pulse combustion. First, the process is inherently noisy. At higher frequencies this is not a major obstacle, since the sound and associated vibrations can then be damped by conventional methods of isolation. And sometimes the sound generated is a plus. For example in drying applications, where sound induced oscillations in a drying chamber not only increase the drying rates, but also prevent depositing of substances on the walls. A more serious problem results from the highly coupled physical and chemical processes in a working pulse combustor, such as the interaction between (turbulent) combustion and the system's acoustics. These interactions cause problems when



(a) Advanced pulse combustor in action (b) Close-up of pulse combustor

Figure 1.1: A gas-fired pulse combustor with a secondary burner, developed by DLF Sustainable. (a) The pulse combustor in operation, the eight tailpipes are glowing. In the secondary burner, pulverized brown coal is ignited by a flame from the pulse combustor's aerovalve. (b) A close-up of the pulse combustor, showing the gas supply at the front, and a flame from the aerovalve entering the secondary burner.

trying to scale up or improve upon a working design. Adjusting the geometry of the pulse combustor may produce the desired effect, but sometimes it may be detrimental to the system's performance instead. Therefore, improvement of a design is mainly a process by trial-and-error.

To aid in the design of pulse combustors and to get a better understanding of the fundamental physical processes that govern the pulse combustion phenomenon, a mathematical model capturing the most important aspects of pulse combustion is desired. The work reported here is a preparing step for research aimed at developing such a mathematical model.

1.2 Objective and outline

The objective of the work presented in this report was threefold. One aim was to get familiarized with the pulse combustion phenomenon, and the mathematical modeling thereof. A second aim was to determine what physical processes or principles are of main importance when describing the pulse combustion phenomenon in mathematical terms. A third aim, in fact the main goal of the study, was to formulate research questions that need to be addressed in the ensuing research phase.

In pursuit of these goals, three simple mathematical models for pulse combustion from literature were studied in detail. These are lumped parameter models: physical processes are modeled by spatial averaging. In this way, modeling of the details of the pulse combustion process is avoided, while one tries to retain the most important characteristics.

This report is organized in the following way. The basic principles of pulse combustion operation are described in Chapter 2. Of special interest is the Rayleigh criterion. It is a condition under which periodic heat release may drive acoustic oscillations, which is a fundamental part of the pulse combustion process. In each of the Chapters 3–5, one simple mathematical model from literature is described and analyzed in detail. Conclusions of the study of these models are presented at the end of the relevant chapters. In Chapter 3 a model presented by Kilicarslan [17] is discussed. Shortcomings of this model to describe a pulse combustion process were reason to study the model of Ahrens et al. [2], that provided the basis for Kilicarslan's model. The model of Ahrens et al. is discussed in Chapter 4. The mathematical model of Richards et al. [26] is the third model that was studied. It includes physical processes that were neglected in the other two models; it is described in Chapter 5. Finally, in Chapter 6 overall conclusions are given, and research questions are formulated that provide a basis for follow-up research. Computational analysis of the three models was performed using Matlab [30] software. The codes that were used, can be found in the appendices.

Chapter 2

Working principles of pulse combustion

The principles of operation of a pulse combustor are based on a coupling between intermittent (pulse) combustion and (resonant) acoustics in the burner system. The interaction of the system's acoustics with the combustion process (e.g. fuel supply, mixing processes, reaction rates, etc.) is such that the resulting intermittent combustion process feeds energy into the acoustical oscillations.

Although the acoustical oscillations in a pulse combustor are obviously driven by the combustion process, the term 'combustion-driven oscillations' is generally associated with disturbances (instabilities) of a steady combustion process. Putnam et al. [23] distinguished between combustion-driven oscillations, which may be harmful in devices designed for steady combustion, and pulse combustion, where oscillations are deliberately stimulated in order to obtain increased mixing rates, higher heat transfer rates, etc. A condition under which unsteady combustion may drive pressure waves, that is often mentioned in the study of combustion instabilities, is Rayleigh's criterion. It will be discussed below, after describing the operation of a typical pulse combustor.

2.1 A typical pulse combustion cycle

Figure 2.1 depicts an idealized cycle of operation of a pulse combustor with an aerodynamic valve (or, aerovalve). In the first stage of the cycle (1), a combustible mixture in the combustion chamber ignites, and a burning region expands, driving gases into the tailpipe (to the right) and through the aerovalve (to the left). In the second stage of the cycle (2), when most of the fuel has been consumed, the gases continue their motion due to inertia, thereby lowering the pressure in the combustion chamber. Eventually, the pressure in the combustion chamber falls below atmospheric pressure. Then the third phase of the cycle starts (3). Fresh air (and fuel) enter the combustion chamber, following the remnant of combustion products from the aerovalve. Due to inertia, the flow in the tailpipe takes longer to slow down and it may reverse its direction at a later moment. In the last stage of the cycle (4), inertia of the gases flowing toward the combustion chamber cause the pressure to rise above the ambient pressure. Then, the cycle is started anew with the ignition of the combustible mixture.

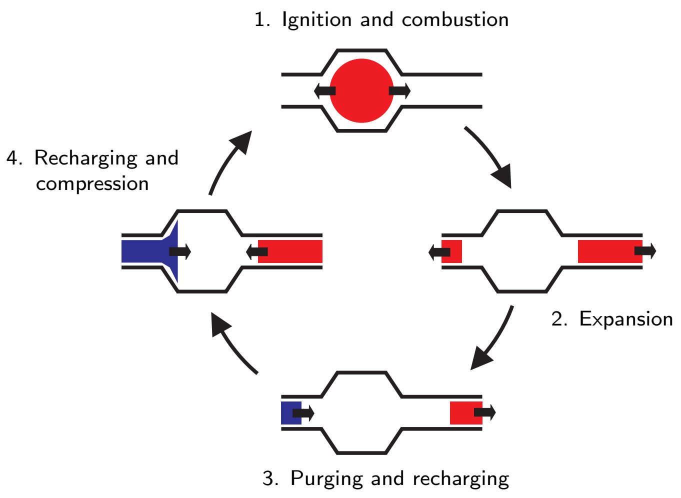

1. Ignition and combustion
4. Recharging and compression
2. Expansion
3. Purging and recharging

Figure 2.1: An idealized (aerovalved) pulse combustion cycle (adapted from Putnam et al. [23]). With a flapper valve, the flow in the direction of the valve is completely blocked during the expansion phase.

The exact ignition mechanism is not clear; it may be caused by flame remnants, contact with hot combustion products, or a hot-spot in the combustion chamber.

There are many different types of pulse combustors. A pulse combustor can be mechanically valved (e.g., it may be equipped with flapper valves or with reed valves), or aerodynamically valved. In case of mechanical valves, back flow through the valve is prevented: flow through the valves only occurs in stages 3 and 4 of Figure 2.1. Also, an aerovalve may be designed to greatly resist back flow, acting somewhat like a badly leaking flapper valve. There is a great variety in the geometry of pulse combustors, too. Typical examples are pulse combustors of the Helmholtz-type, the Schmidt (or, quarter-wave) type, and the Rijke-type. The pulse combustor sketched in Figure 2.1 is of the Helmholtz-type: a volume (the combustion chamber) acts like a Helmholtz resonator on the tailpipe gases. A Schmidt-type pulse combustor typically consists of a tube closed at one end (where combustion takes place) and open at the other end; it acts like a quarter-wave tube. In a Rijke-type pulse combustor, combustion takes place in the lower half of a vertical tube, acting like a Rijke-tube. In this configuration, the combustion process is typically enhanced by (acoustical) velocity fluctuations. Other configurations are possible. In a review article on pulse combustion, Putnam et al. [23] provide a historic overview of pulse combustor designs, and describe the operating characteristics of mechanically valved and aerovalved pulse combustors.

2.2 Rayleigh's criterion

A pulse combustor can be viewed as a harmonic resonator used to store acoustical energy. Moments of low pressure are used for refueling, while periodic combustion sustains the acoustical oscillations by feeding energy to the resonator. A widely used condition under which periodic heat release (e.g. by combustion) can sustain acoustical oscillations, is Rayleigh's criterion. In short, it states that acoustical oscillations are stimulated when heat is released in phase with the pressure variations, while the oscillations are damped when heat is released out of phase.

In 1878, Rayleigh [24] formulated it as follows:

> "If heat be periodically communicated to, and abstracted from, a mass of air vibrating (for example) in a cylinder bounded by a piston, the effect produced will depend upon the phase of the vibration at which the heat transfer takes place. If heat be given to the air at the moment of greatest condensation, or taken from it at the moment of greatest rarefaction, the vibration is encouraged. On the other hand, if heat be given at the moment of greatest rarefaction, or abstracted at the moment of greatest condensation, the vibration is discouraged."

He also noted that the frequency can be changed, depending on the phase of heat release:

> "If the air be at its normal density at the moment when the transfer of heat takes place, the vibration is neither encouraged nor discouraged, but the pitch is altered. Thus the pitch is raised, if heat be communicated a quarter period before the phase of greatest condensation, and the pitch is lowered if the heat be communicated a quarter period after the phase of greatest condensation."

Rayleigh remarked that in general heat release produces both kinds of effects. He went on to illustrate the principle by successfully applying it to explain certain acoustical phenomena.

2.2.1 A heuristic explanation

Intuitively, Rayleigh's criterion is easily understood. For example, consider a volume of air vibrating in a surrounding fluid. Energy of condensation of the volume (i.e. its potential energy) and kinetic energy of the surroundings are exchanged through work performed on the boundaries of the volume. Heat addition will cause the volume of air to expand, obtaining a new equilibrium volume. When heat is added to the volume at a moment of condensation, the action of the volume of air is supported in converting potential energy into kinetic energy (or vice versa), and the vibration is encouraged. On the other hand, when heat is added at a moment of rarefaction, the action of the volume of air on the surroundings is counteracted, and the vibration is discouraged. The greater the pressure difference of the volume and the surroundings, the more work is done per unit of expansion. Thus, the effect of heat addition is greater if the condensation is greater. When the volume of air is at its equilibrium pressure, it is not doing any work on the surroundings, and heat addition will neither encourage nor discourage the vibration.

The effect of (periodic) heat release on the frequency of the vibration can be understood intuitively, too. If heat is added before the moment of greatest condensation, the induced expansion works against the compression by the surroundings, and the state of greatest condensation is reached sooner. Therefore, the frequency is increased. If heat is added after the moment of greatest condensation, the induced expansion works with the expansion by the surroundings, and the state of greatest rarefaction is reached later. Therefore, the frequency is decreased.

2.2.2 A mathematical formulation

An often used mathematical formulation of the Rayleigh criterion for maintenance of oscillations is

$$\int_0^{T_P} \int_V p'Q'\,dV\,dt > 0, \tag{2.1}$$

where the product of the pressure fluctuations p' and the fluctuations in heat release Q' is integrated over the volume V and the time interval $[0, T_P]$, with T_P a period of oscillation. It can be viewed as an expression of the heuristic argument put forward in the previous subsection. More generally, in order for the heat release to provide the oscillations with sufficient energy to overcome energy losses, the integral on the left-hand side should be greater than the sum of the energy losses ($\sum_i L_i$) occurring in the volume over a period of oscillation:

$$\int_0^{T_P} \int_V p'Q'\,dV\,dt > \sum_i L_i. \tag{2.2}$$

Energy losses may be associated with viscous dissipation, heat conduction, energy fluxes through the boundary of the volume, etc. This is a generalized version of the Rayleigh criterion.

In an article on the energy transfer to small disturbances in fluid flow, Chu [6] has provided a foundation for expressions like (2.1) and (2.2), quantifying the conditions under which heat release encourages fluctuations in fluid flow. In this report, no quantification of the Rayleigh criterion is used, therefore the article of Chu will not be discussed at length here. However, to illustrate how the Rayleigh criterion may appear from the general equations of fluid motion, a simplified derivation may be insightful. Such a derivation will be considered next. For details on the derivation, see the paper by Chu [6].

Consider a uniform homogeneous perfect gas at rest: its state may be expressed by velocity $\mathbf{u}_0 = \mathbf{0}$, pressure $p = p_0$, density $\rho = \rho_0$ and temperature $T = T_0$, all constant. Assume that the specific heats for constant pressure c_p and constant volume c_v are constant, and that effects of viscosity and heat conduction may be neglected. Now introduce some small heat release Q', and associated *small* disturbances in velocity ($\mathbf{u}'$), pressure (p'), density (ρ'), and temperature (T'). The disturbances are considered small if: $|\mathbf{u}'|/c_0 \ll 1$ (with c_0 the speed of sound of the undisturbed fluidum), $|p'/p_0| \ll 1$, $|\rho'/\rho_0| \ll 1$, and $|T'/T_0| \ll 1$. Now, substitution of $p = p_0 + p'$, $\mathbf{u} = \mathbf{u}_0 + \mathbf{u}'$, etc., in the continuity equation, the momentum equation, the energy equation, and the state equation, and neglecting cross-products of fluctuations, leads to the following equations for the disturbances in the fluid flow:

$$\frac{\partial \rho'}{\partial t} + \rho_0 \nabla \cdot \mathbf{u}' = 0, \qquad \text{(continuity)} \tag{2.3}$$

$$\rho_0 \frac{\partial \mathbf{u}'}{\partial t} = -\nabla p', \qquad \text{(momentum)} \tag{2.4}$$

$$\rho_0 c_v \frac{\partial T'}{\partial t} + p_0 \nabla \cdot \mathbf{u}' = Q', \qquad \text{(energy)} \tag{2.5}$$

$$\frac{T'}{T_0} + \frac{\rho'}{\rho_0} = \frac{p'}{p_0}. \qquad \text{(state)} \tag{2.6}$$

To see how the heat release Q' drives the pressure disturbances, substitute T' from the state equation (2.6) into the energy equation (2.5), and then write the time-derivative of ρ' in terms of the rate of expansion $(\nabla \cdot \mathbf{u}')$ using the continuity equation (2.3). Using $\gamma = c_p/c_v$ and the perfect gas properties $p_0 = R\rho_0 T_0$ and $R = c_p - c_v$, with R the gas constant, this gives

$$\frac{1}{\gamma p_0}\frac{\partial p'}{\partial t} + \nabla \cdot \mathbf{u}' = \frac{Q'}{c_p \rho_0 T_0}. \tag{2.7}$$

Differentiating this equation with respect to time, and eliminating the time derivative of the rate of expansion $(\frac{\partial}{\partial t}\nabla \cdot \mathbf{u}')$ by substitution of the momentum equation (2.4) after taking the divergence, leads to

$$\frac{\partial^2 p'}{\partial t^2} - c_0^2 \nabla^2 p' = (\gamma - 1)\frac{\partial Q'}{\partial t}, \tag{2.8}$$

where the relation $c_0^2 = \gamma p_0/\rho_0$ for speed of sound has been used. This is an inhomogeneous wave equation, with the time-derivative of the heat release rate as source term.

An expression for the acoustic energy can be derived from the momentum equation equation (2.4) and equation (2.7), leading to a quantification of the Rayleigh criterion. Adding the product of p' and equation (2.7) to the inner product of $\mathbf{u}'$ and equation (2.4), and using $\rho_0 c_0^2 = \gamma p_0$ and $\frac{1}{c_p \rho_0 T_0} = \frac{\gamma - 1}{\gamma p_0}$, yields

$$\frac{\partial}{\partial t}\left[\frac{1}{2}\frac{p'^2}{\rho_0 c_0^2} + \frac{1}{2}\rho_0 \mathbf{u}' \cdot \mathbf{u}'\right] + \nabla \cdot (p'\mathbf{u}') = \frac{\gamma - 1}{\gamma p_0} p' Q'. \tag{2.9}$$

This is an equation for the acoustic energy: the two terms in the square brackets are the energy associated with compression (first term, a potential energy) and the kinetic energy (second term), and the divergence on left-hand side expresses the flux of acoustic energy. The right-hand side shows a source term for the production of acoustic energy. Integrated over a volume V with boundary S and a period of oscillation $[0, T_P]$, this equation provides a quantification of Rayleigh's criterion: in order for the acoustic disturbances to be maintained, the relation between the pressure fluctuations p' and heat release Q' should satisfy the inequality

$$\frac{\gamma - 1}{\gamma p_0}\int_0^{T_P}\int_V p' Q' \, dV \, dt > \int_0^{T_P}\int_S p' \mathbf{u}' \cdot \mathbf{n} \, dS \, dt, \tag{2.10}$$

where $\mathbf{n}$ denotes the outwardly directed normal on the surface S. To convert the volume integral of the divergence to the surface integral, Gauss' theorem has been used.

In his article, Chu [6] included viscous and heat conduction terms, source terms for mass and heat, and body forces. Taking the fluxes across the boundary of the volume of interest equal to zero, he derived an expression that generalizes the acoustic energy of the fluctuations to an expression including the fluctuations in entropy. If heat conduction can be neglected, and without mass sources and body forces, his expression for the rate of change of the energy of fluctuations splits into two distinct expressions:

$$\frac{\partial}{\partial t}\int_V \left[\frac{1}{2}\frac{p'^2}{\rho_0 c_0^2} + \frac{1}{2}\rho_0 \mathbf{u}' \cdot \mathbf{u}'\right] dV = \frac{(\gamma-1)}{\gamma p_0}\int_V p'Q'\,dV - \int_V \Phi'\,dV, \qquad (2.11)$$

where Φ' is a viscous dissipation term, and

$$\frac{\partial}{\partial t}\int_V \left[\frac{1}{2}\frac{(\gamma-1)}{\gamma}p_0\left(\frac{S'}{R}\right)^2\right] dV = \frac{1}{c_p}\int_V S'Q'\,dV, \qquad (2.12)$$

where S' is the fluctuation in entropy. The last equation can be obtained directly from the energy equation if it is expressed in terms of entropy instead of temperature: $\rho_0 T_0 \frac{\partial S'}{\partial t} = Q'$. Based on equation (2.11), the Rayleigh criterion may be expressed as

$$\int_0^{T_P}\int_V p'Q'\,dV\,dt > \frac{\gamma p_0}{(\gamma-1)}\int_0^{T_P}\int_V \Phi'\,dV\,dt, \qquad (2.13)$$

but equation (2.12) shows that the Rayleigh criterion does not tell the whole story. If heat is released in phase with the entropy fluctuations, the flow may become unstable, even without any pressure fluctuations. If heat conduction cannot be neglected, the contributions of source terms to the energy of fluctuations cannot be split into an acoustical part and an entropic part, as was done with (2.11) and (2.12).

Further, Chu [6] applied the theory of the energy of fluctuations to study transfer of energy from a steady main stream, with extra source terms derived from the main stream. He then discussed the special case of a two-dimensional parallel shear flow as main stream, showing the work done by the Reynolds stress and the transport of entropy spottiness across the temperature shear layers as production terms of the energy of fluctuations.

In a closely related article, Nicoud and Poinsot [20] first derived an exact nonlinear equation, similar to the equation for the energy of fluctuations. From this, they derived an equation for the energy of fluctuations by linearization. They mentioned an extra production term for the energy of fluctuations, that appears to be missing in the derivation by Chu: $-\frac{p_0}{RC_p}S'\mathbf{u}' \cdot \nabla S_0$, where S_0 is the entropy of the mean flow. After an estimation of the order of magnitude of the extra term, they noted that it may be larger that the classic Rayleigh term. They concluded that for the investigation of combustion instabilities, the energy of fluctuations should be used instead of Rayleigh's criterion.

Chapter 3

A model suggested by Kilicarslan

Note: In a Bachelor's thesis project at the Eindhoven University of Technology, Tiemessen and Bastiaans [32] adjusted the model of Kilicarslan [17], which is discussed in this chapter, with the purpose of predicting the frequency of an experimental aerovalved pulse combustor. This report was studied, but the discussion in this chapter focuses only on the work presented by Kilicarslan.

In this chapter a zero-dimensional model for a gas-fired flapper-valved pulse combustor of the Helmholtz-type, presented in an article by Kilicarslan [17], will be examined. Figure 3.1 shows a schematic of the modeled pulse combustor. In the model, the combustion chamber is taken as control volume and an energy balance for this control volume is formulated. The amount of energy in the control volume is changed by reactants mixture entering the control volume, by combustion inside the control volume and by products leaving the control volume via the tailpipe. The inflow of reactants mixture is modeled by quasi-steady flow orifice equations and depends upon the pressure difference between the combustion chamber pressure and the supply pressure (assumed to be atmospheric). The tailpipe flow is modeled as a plug flow, i.e., of constant density and solely driven by the pressure difference over the tailpipe. The heat release by combustion is assumed to be equal to the heat of combustion of the reactants mixture entering the combustion chamber. No energy losses, e.g., by viscous effects or heat transfer, are modeled. The gases are assumed to be perfect. The specific heats of the reactants entering the combustion chamber, of the gases inside it, and of the products leaving it to the tailpipe, are assumed to be constant and all equal. Apparently, some kind of average value is used. This is also assumed for the gas constants of the reactants, the products, and the mixture inside the combustion chamber.

From the mathematical model, Kilicarslan derived a frequency of operation for the pulse combustor. In the article, the theoretical frequency of operation is compared to experimental results with a tunable pulse combustor. In the experiments, the tailpipe length, tailpipe diameter, combustion chamber volume and supply pressure were varied.

Below, the derivation of the mathematical model of Kilicarslan is given,

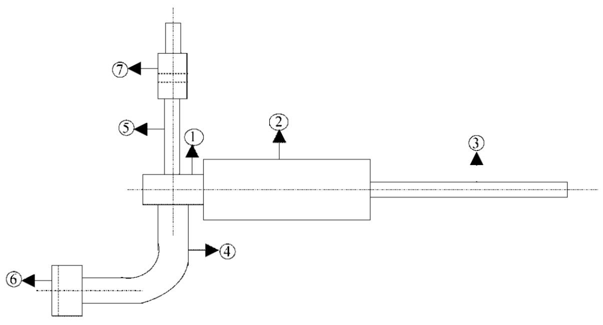


Figure 3.1: A schematic of the modeled gas-fired pulse combustor. (1) Mixer head, (2) combustion chamber, (3) tailpipe, (4) air supply, (5) gas supply, (6,7) flapper valves. The figure is taken from the article by Kilicarslan [17].

largely following Kilicarslan's presentation in [17]. The nomenclature is changed to ease the comparison with the other models examined in this report. As will be seen, the model for the heat release implies a timing of heat release such that the oscillatory motions necessary for pulse combustion operation are damped. Therefore, this model is not a proper description of a working pulse combustor. In the derivation of the theoretical frequency, Kilicarslan assumed that the damping that results from the (incorrect) modeling of the heat release is negligible. The fact that the Rayleigh criterion is not satisfied, which is an indication that the model does not correctly describe a pulse combustion process, manifests itself in the (neglected) damping term. This may be the reason why the incorrect modeling of the heat release has gone unnoticed.

3.1 Nomenclature

Roman symbols

A–C	Auxiliary constants, see (3.18)–(3.20)	
A_{tp}	Area of tailpipe cross-section	$[\mathrm{m}^2]$
A_v	Effective flow area air/gas valve	$[\mathrm{m}^2]$
C_D	Discharge coefficient of valve	$[-]$
c_p	Specific heat for constant pressure	$[\mathrm{J/kg\,K}]$
c_v	Specific heat for constant volume	$[\mathrm{J/kg\,K}]$
e	Specific internal energy	$[\mathrm{J/kg}]$
E	Total energy	$[\mathrm{J}]$
f	Frequency	$[\mathrm{Hz}]$
F	Auxiliary function, see (3.21)	
h	Specific enthalpy	$[\mathrm{J/kg}]$
k_g	Gas valve parameter	$[\mathrm{kg}^{1/2}\,\mathrm{m}^{1/2}]$
L_{tp}	Tailpipe length	$[\mathrm{m}]$
$\dot{m}$	Mass flux	$[\mathrm{kg/s}]$

Continued on next page

Roman symbols, continued

p	Gauge pressure (in combustion chamber)	[Pa]
P	Pressure (in combustion chamber)	[Pa]
$\dot{Q}$	Rate of heat release	[J/s]
r	Air-fuel mass ratio	[−]
R	Specific gas constant	[J/kg K]
t	Time	[s]
T	Temperature	[K]
V	Volume	[m^3]

Greek symbols

ΔH_f	Heat of combustion per unit fuel mass	[J/kg]
γ	Ratio of specific heats, $= c_p/c_v$	[−]
ω_0	Angular frequency	[rad/s]
ρ	Density	[kg/m^3]

Subscripts

0	initial/standard/ambient value
a	air (oxidizer)
cc	combustion chamber
e	exit combustion chamber/entrance tailpipe
g	gas (fuel)
p	products (burnt reactants)
r	reactants (fuel and oxidizer)
tp	tailpipe

3.2 Derivation of model equations

The combustion chamber is chosen as a control volume. See Figure 3.2 for a schematic of the control volume and sign conventions. Reactants entering the combustion chamber consist of premixed air and gas with a constant air-fuel ratio r (on mass basis). All reactants are assumed to be converted to products by combustion. The products leave the combustion chamber through the tailpipe.

The gases are assumed to be perfect, i.e., (i) they obey the ideal gas law:

$$P = \rho R T, \tag{3.1}$$

where P is the pressure, ρ the density, R the specific gas constant and T the temperature of the gas, and (ii) the ratio of specific heats, γ, is constant:

$$\gamma = \frac{c_p}{c_v} = \text{constant}, \tag{3.2}$$

where c_p and c_v are the specific heat for constant pressure and volume, respectively. Since the molecular weight is constant the specific gas constant does not vary in absence of chemical reaction. In that case the specific heats are both constant and the specific internal energy and specific enthalpy can be written as

$$e = c_v T, \tag{3.3}$$

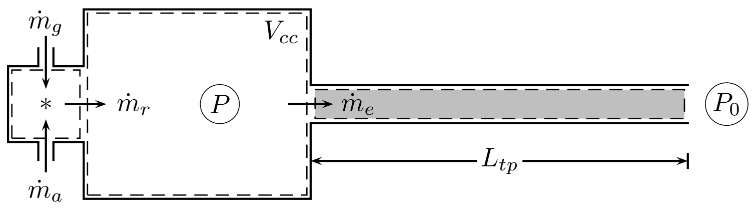


Figure 3.2: Control volume (i.e., the combustion chamber, with volume V_{cc}) of the modeled pulse combustor, with sign conventions and (some) model parameters. The values of the mass fluxes $\dot{m}_g$, $\dot{m}_a$, $\dot{m}_r$ and $\dot{m}_e$ are taken positive in the directions indicated. Mixing of gas and air takes place in the mixer head ($*$) and the reactants entering the combustion chamber are assumed to be perfectly mixed.

and

$$h = c_p T, \tag{3.4}$$

respectively. Kilicarslan assumes that the ratios of the specific heats for constant volume to the specific gas constants are equal for reactants and products, thus

$$\left(\frac{c_v}{R}\right)_r = \left(\frac{c_v}{R}\right)_p = \frac{c_v}{R} = \frac{1}{\gamma - 1} = \text{constant}. \tag{3.5}$$

The reactants occupy a volume V_r of the combustion chamber and the products occupy a volume V_p. The combustion chamber is assumed to be fully occupied at all times, thus, with V_{cc} denoting the volume of the combustion chamber:

$$V_r + V_p = V_{cc}. \tag{3.6}$$

The state variables of the reactants and products are assumed to be uniform throughout their volumes and the pressures of the reactants, P_r, and products, P_p, associated with the volumes V_r and V_p, respectively, are assumed to be equal at all times:

$$\rho_r R_r T_r = P_r = P = P_p = \rho_p R_p T_p. \tag{3.7}$$

3.2.1 Conservation of energy

An energy balance over the control volume (i.e., the combustion chamber) yields

$$\frac{dE_{cc}}{dt} = \dot{Q} + h_r \dot{m}_r - h_e \dot{m}_e, \tag{3.8}$$

where E_{cc} is the total energy in the combustion chamber, $\dot{Q}$ is the rate of heat release by combustion, $\dot{m}_r$ and $\dot{m}_e$ are the mass fluxes into and out of the combustion chamber, respectively, and h_r and h_e are the specific (sensible) enthalpies[1] of the reactants and products, respectively. Under the assumptions

[1] A standardized (or, absolute) enthalpy (per unit mass) $h(T)$ at a temperature T can be written as $h(T) = h_f^\circ(T_{\text{ref}}) + \Delta h_s(T)$, where $h_f^\circ(T_{\text{ref}})$ is the enthalpy of formation of the substance (associated with the chemical bonds) at a certain reference temperature T_{ref} and $\Delta h_s(T)$ is the sensible enthalpy change, i.e., the difference in enthalpy when going from the reference temperature T_{ref} to the temperature T: $\Delta h_s = \int_{T_{\text{ref}}}^{T} dh$. The heat of combustion

mentioned above the total energy in the combustion chamber can be written as

$$E_{cc} = \rho_r e_r V_r + \rho_p e_p V_p = \frac{c_v}{R}(P_r V_r + P_p V_p) = \frac{P V_{cc}}{\gamma - 1}. \tag{3.9}$$

The rate of heat release $\dot{Q}$ in the combustion chamber is given by

$$\dot{Q} = \dot{m}_r \frac{\Delta H_f}{1+r} = \dot{m}_r (h_e - h_r), \tag{3.10}$$

where ΔH_f is the heat of combustion per unit mass of fuel.

3.2.2 Mass flow rate into combustion chamber

The mass flow rate for the reactants (gas and air separately) are modeled by quasi-steady orifice flow equations, assuming the flapper valves to be either fully open or fully closed. In order to keep the air-fuel ratio, r, constant the pressure at the gas-supply is assumed to be equal to the atmospheric pressure, P_0, at which the air is supplied. This yields for the time-dependent mass flow rate of gas, $\dot{m}_g$,

$$\dot{m}_g(t) = k_g \sqrt{|p|}\, \mathbf{1}_{(-\infty,0)}(p), \tag{3.11}$$

where k_g is a constant parameter corresponding to the flow through the gas valve, p is the gauge pressure of the combustion chamber, i.e., $p = P - P_0$, and $\mathbf{1}_X(x)$ is an indicator function equal to one if x is an element of the set $X \subset \mathbf{R}$ and zero otherwise. Thus, there is only inflow if the pressure in the combustion chamber is below the atmospheric pressure. Since $\sqrt{|p|} = \sqrt{-p}$ for $p < 0$, expression (3.11) could equivalently be written with $-p$ instead of the absolute value $|p|$. The form given in (3.11) is preferred in order to avoid confusion about the meaning of $\sqrt{-p}$ for $p > 0$. The constant k_g is given by

$$k_g = \sqrt{2\rho_g} C_{D,g} A_{v,g}, \tag{3.12}$$

where ρ_g is the density of the gas, $C_{D,g}$ is a discharge coefficient of the gas valve and $A_{v,g}$ is the effective flow area of the gas valve (see Ahrens et al. [2]).

For the mass flow rate of air, $\dot{m}_a$, a similar expression can be written with corresponding parameters ρ_a, $C_{D,g}$ and $A_{v,a}$. Using the air-fuel ratio, r, the mass flow rate of the reactants can be expressed as $\dot{m}_r = \dot{m}_g(1+r)$, thus

$$\dot{m}_r(t) = k_g (1+r) \sqrt{|p|}\, \mathbf{1}_{(-\infty,0)}(p). \tag{3.13}$$

Note that for $p < 0$ the air-fuel ratio is given by $r = \dot{m}_a / \dot{m}_g$, yielding

$$r = \sqrt{\frac{\rho_a}{\rho_g}} \frac{C_{D,a} A_{v,a}}{C_{D,g} A_{v,g}}. \tag{3.14}$$

(Δh_f) at a certain reference state is defined as the difference of the standardized enthalpies of the reactants (h_{reac}) and the products (h_{prod}) at that reference state and, thus, is equal to the difference in enthalpies of formations at that reference state: $\Delta h_f(T_{\text{ref}}) = h_{\text{reac}}(T_{\text{ref}}) - h_{\text{prod}}(T_{\text{ref}}) = h^{\text{o}}_{f,\text{reac}}(T_{\text{ref}}) - h^{\text{o}}_{f,\text{prod}}(T_{\text{ref}})$. In the derivation of the model equations, the enthalpies of formation are contained in the heat release term, so the enthalpies h_r and h_e in the derivation are sensible enthalpies, or sensible enthalpy changes. See, for example, [33, p.24–31] for details.

3.2.3 Coupling to tailpipe dynamics by conservation of momentum

The flow in the tailpipe is assumed to behave like a plug flow: it is considered to be of constant density and moving frictionlessly under the influence of the pressure difference between tailpipe inlet and exit. Conservation of momentum in the tailpipe gives the following differential equation for the mass flow rate $\dot{m}_e$ into the tailpipe:

$$L_{tp}\frac{d\dot{m}_e}{dt} = pA_{tp}, \tag{3.15}$$

where L_{tp} is the tailpipe length and A_{tp} the area of its cross-section.

3.2.4 Complete set of model equations

Substitution of the total energy (3.9), the rate of heat release (3.10) and the mass flow rate of reactants (3.13) into the energy balance equation (3.8) together with equation (3.15) for conservation of momentum lead to the following system of first-order nonlinear differential equations

$$\frac{dp}{dt} = AF(p) - B\,\dot{m}_e, \tag{3.16}$$

$$\frac{d\dot{m}_e}{dt} = C\,p, \tag{3.17}$$

where

$$A = k_g(1+r)\frac{(\gamma-1)h_e}{V_{cc}}, \tag{3.18}$$

$$B = \frac{(\gamma-1)h_e}{V_{cc}}, \tag{3.19}$$

$$C = \frac{A_{tp}}{L_{tp}}, \tag{3.20}$$

$$F(p) = \sqrt{|p|}\,\mathbf{1}_{(-\infty,0)}(p), \tag{3.21}$$

with (see expression (3.10))

$$h_e = \left(h_r + \frac{\Delta H_f}{1+r}\right). \tag{3.22}$$

By differentiating equation (3.16) and substitution of equation (3.17) a single second-order nonlinear differential equation is derived. Note that the function $F(p)$ is not differentiable in $p = 0$. Thus, its derivative $F'(p)$ is not properly defined in that point; it will be defined as $F'(0) \equiv 0$, i.e., equal to the right-hand derivative of F in $p = 0$. This yields the model equation of Kilicarslan for the gauge pressure p in the combustion chamber:

$$\frac{d^2p}{dt^2} - AF'(p)\frac{dp}{dt} + BCp = 0, \tag{3.23}$$

where

$$F'(p) = \frac{dF}{dp} = -\frac{1}{2\sqrt{|p|}}\,\mathbf{1}_{(-\infty,0)}(p). \tag{3.24}$$

The mathematical model is completed by supplying appropriate initial conditions, e.g., prescribing the initial gauge pressure, $p(0) = p_0$, and initial mass flux out of the combustion chamber, $\dot{m}_e(0) = \dot{m}_{e,0}$.

3.3 Prediction of frequency of pulse combustion

Kilicarslan uses equation (3.23) to estimate the frequency of a stable pulse combustion process. Without giving much justification it is implicitly assumed that the damping term $-AF'(p)\frac{dp}{dt}$ of equation (3.23) is of negligible effect on the frequency of the pulse combustion process. In that case the approximated frequency of the simulated combustion process, f_0, is given by

$$f_0 = \frac{\omega_0}{2\pi} = \frac{1}{2\pi}\sqrt{BC} = \frac{\sqrt{(\gamma-1)h_e}}{2\pi}\sqrt{\frac{A_{tp}}{V_{cc}L_{tp}}}, \tag{3.25}$$

where ω_0 is the angular frequency associated with the harmonic oscillator described by ignoring the damping term $-AF'(p)\frac{dp}{dt}$ in equation (3.23). Note that this is simply the frequency of a Helmholtz resonator without resistance in the case that the speed of sound is given by that in the tailpipe: $\sqrt{\gamma RT_e} = \sqrt{(\gamma-1)c_pT_e} = \sqrt{(\gamma-1)h_e}$, where T_e is the temperature in the tailpipe, i.e., at the exit of the combustion chamber. (See Thompson [31, pp. 549–550] for a derivation of the resonant frequency of a Helmholtz resonator.)

3.4 Comments on the mathematical model

3.4.1 Damping effects

The model equation (3.23) can be written as

$$\ddot{x} + h(x,\dot{x}) + \omega_0^2 x = 0, \tag{3.26}$$

where $x = p(t)$, ω_0 is given by expression (3.25), a dot denotes a time derivative and $h(x,\dot{x})$ is defined by

$$h(x,\dot{x}) = -AF'(x)\dot{x}, \tag{3.27}$$

with $F'(x)$ given by (3.24). The equation can be interpreted as describing a linear oscillator with nonlinear damping. Following the analysis of Jordan and Smith [13, pp. 16–17], we can write the total mechanical energy $\mathcal{E}$ of the oscillator as

$$\mathcal{E} = \mathcal{T} + \mathcal{V} = \frac{1}{2}\dot{x}^2 + \frac{1}{2}\omega_0^2 x^2, \tag{3.28}$$

where $\mathcal{T} = \frac{1}{2}\dot{x}^2$ and $\mathcal{V} = \frac{1}{2}\omega_0^2 x^2$ are the kinetic energy and a potential energy, respectively. Along a phase path $(x(t),\dot{x}(t))$ the energy variation with time is given by

$$\frac{d\mathcal{E}}{dt} = \frac{d\mathcal{T}}{dt} + \frac{d\mathcal{V}}{dt} = \dot{x}\ddot{x} + \omega_0^2 x\dot{x} = -h(x,\dot{x})\dot{x}. \tag{3.29}$$

Integration of this expression with respect to time from t_0 to t_1 gives

$$\mathcal{E}(t_1) - \mathcal{E}(t_0) = -\int_{t_0}^{t_1} h(x,\dot{x})\dot{x}dt. \tag{3.30}$$

Because $h(x,\dot{x})\dot{x} = -AF'(x)\dot{x}^2 \geq 0$ for all times (and strictly so if $x < 0$) the energy will be decreasing as time passes. The amplitudes of oscillation in x, i.e., in the gauge pressure p in the combustion chamber, will be diminishing in time and never increasing. Thus, the mathematical model presented by Kilicarslan cannot describe a stable operating pulse combustor.

3.4.2 Using Rayleigh's criterion to explain damping effect

Above it was found that the amount of damping (in the sense of energy loss per unit of time) is related to $-AF'(p)(\frac{dp}{dt})^2 \geq 0$, which is proportional to the parameter A. This parameter is given by expressions (3.18) and (3.22) and it comes from mass flow of reactants into the combustion chamber (part of the sum in A proportional to h_r, see (3.22)) and from heat release in the combustion chamber (part of A proportional to ΔH_f, see (3.22)). On noting that the mass flow rate into the combustion chamber occurs when the gauge pressure is negative and the rate of heat release is modeled to be proportional to the mass flow rate an explanation for the damping is given by application of the Rayleigh's criterion. At the moment of greatest *rarefaction* both mass and heat are added to the combustion chamber, both with the effect of diminishing the pressure amplitude. In pulse combustors the heat release should be greatest at moments of greatest *condensation* thereby amplifying the oscillatory motion.

Clearly the heat release is modeled incorrectly in the work of Kilicarslan. His mathematical model for pulse combustion is largely based on that of Ahrens et al. [2]. The main difference is in the modeling of the heat release. That is why in the next chapter the model of Ahrens et al. will be examined.

3.5 Numerical implementation and experiments

The system of model equations (3.16)–(3.22) is numerically implemented using Matlab [30] software. The standard Matlab ODE-solver `ode45`[2] is used to solve the equations. The implementation in Matlab is straightforward; the codes can be found in Appendix A.

In order to solve the model equations numerically, the model parameters need to be specified, as well as the initial conditions. For the purpose of showing the damping of the pressure fluctuations in the mathematical model, one or two sets of parameters will suffice. In the experiments of Kilicarslan, the gas supply pressure and geometrical properties of the combustor (the volume of the combustion chamber, and the tailpipe length and diameter) were varied. For the numerical evaluation of the model equations, the gas supply pressure will be chosen equal to the air supply pressure of 1 atm. The other parameters that were varied in the experiments are listed in Table 3.1, along with ranges over which they were varied and the values that were selected for the numerical evaluation of the model equations. These parameters are chosen such that they lie somewhere in the middle of the intervals over which they were varied.

Except for the gas and air supply pressure (the latter equal to 1 atm), none of the other model parameters were specified by Kilicarslan. Nor is the kind of gas mentioned that was used in the experiments. Therefore, for the numerical evaluation of the model equations, the model parameters are assigned rather arbitrary, although reasonable, values. In order to obtain reasonable values, the article of Ahrens, et al. [2] (upon which a large part of Kilicarslan's analysis is based) was consulted. The heat of combustion of the fuel used for the numerical

[2]The Matlab function `ode45` is part of a suite of ODE-solvers. It is an implementation of a one-step method with adaptive step-size control that uses two explicit Runge-Kutta methods, based on the Dormand-Prince (4,5) pair. That is, it uses fifth and fourth order Runge-Kutta formulae to obtain an estimate for the local error in order to determine the step-size. See [29] for a comprehensive description of the Matlab ODE suite.

Parameter	Range in experiments	Value used	Unit
D_{tp}	$0.038 - 0.051$	0.042	m
L_{tp}	$1 - 3$	2	m
V_{cc}	1.739800×10^{-3} 2.609705×10^{-3} 3.479607×10^{-3}	2.609705×10^{-3}	m^3

Table 3.1: Model parameters connected to the shape of the pulse combustor, the ranges over which the values of these parameters are varied in the experiments of Kilicarslan, and the values used in the numerical evaluation of Kilicarslan's model equations.

evaluation is that of methane (CH_4). The reference temperature and pressure for the heat of combustion are 298.15 K and 1 atm, respectively. The lower heating value (LHV) is used, since part of the heat released is used to evaporate formed water from the reference temperature to the temperature of the products. The discharge coefficients of the valves are chosen the same as in the article of Ahrens et al., who, lacking information for a correct value, chose rather arbitrarily the value 0.6. Air property values (at 300 K and 1 atm) are used for the gas constant and the heat capacity at constant pressure. The value used for the effective flow area of the gas valve is based on that of a pulse combustor of comparable dimensions mentioned in the article of Ahrens et al., and is about 1/100 of the value used for the area of the tailpipe cross-section. Thus, the gas valve diameter is in the order of a tenth of the diameter of the tailpipe. It is assumed that the dimensions of the air valve are such that the air-fuel ratio, r, is constant at the stoichiometric value. This ratio is easily calculated by using the common approximation of air as consisting of 21% oxygen (O_2) and 79% nitrogen (N_2) (by volume), i.e., for each mole of O_2 there is $\frac{79}{21} \approx 3.76$ moles of N_2 in air. Then, using the chemical balance

$$\mathrm{CH_4} + 2\,(\mathrm{O_2} + 3.76\,\mathrm{N_2}) \quad \longrightarrow \quad \mathrm{CO_2} + 2\,\mathrm{H_2O}\ (g) + 7.52\,\mathrm{N_2}, \tag{3.31}$$

the air-fuel ratio (on mass basis) is given by

$$r = \frac{2\,(\mathrm{MW_{O_2}} + 3.76 \times \mathrm{MW_{N_2}})}{\mathrm{MW_{CH_4}}} = \frac{2\,(31.999 + 3.76 \times 28.013)}{16.043} \approx 17.12, \tag{3.32}$$

where $\mathrm{MW_X}$ denotes the molecular weight of a substance X. The values of the molecular weights are obtained from [28, Table 2, p.343].

Some auxiliary constants are needed to compute the model parameters. All values used in the numerical evaluation of the model equations are listed in Table 3.2 (in addition to Table 3.1).

The initial conditions are chosen as indicated in Table 3.3. That is, the initial gauge pressure is equal to the atmospheric pressure (thus, the initial pressure in the combustion chamber is twice the atmospheric pressure) and the initial mass flux into the tailpipe is zero, making the initial pressure a (local) maximum.

The pressure development in the combustion chamber over time predicted by the model equations is shown in Figure 3.3. The figure shows such strong damping that not even one cycle is completed. In order to get some kind of oscillations into the system, the model parameters are adjusted. Oscillation can

Parameter	Value used	Unit	Comment
A_{tp}	$= \pi D_{tp}^2/4$	m^2	
$A_{v,g}$	6.26×10^{-5}	m^2	Ahrens, et al.
$C_{D,g}$	0.6	–	Ahrens, et al.
c_p	1009.7	J/kg K	air prop., [28, p.346]
h_e	$= h_r + \frac{\Delta H_f}{1+r}$	J/kg	
h_r	$= c_p T_0$	J/kg	
k_g	$= \sqrt{2\rho_g} C_{D,g} A_{v,g}$	$\mathrm{kg}^{1/2}\,\mathrm{m}^{1/2}$	
MW_g	16.043	kg/kmol	mol. weight, [28, p.343]
P_0	1.01325×10^5	Pa	$= 1\,\mathrm{atm}$
r	17.12	–	stoich. ratio, computed
R_a	287	J/kg K	air prop., [28, p.343]
R_g	$= R_u/\mathrm{MW}_g$	J/kg K	
R_u	8.31441×10^3	J/kmol K	univ. gas const., [28, in back]
T_0	300	K	
ΔH_f	50.016×10^6	J/kg	LHV, [33, p.543]

Table 3.2: Model parameters (and auxiliary values to compute model parameters) used for the numerical evaluation of Kilicarslan's model equations. Parameters are taken from work of Ahrens et al. [2], computed, or looked up in tables in [28] or [33].

Initial condition	Value used	Unit
$p(0)$	1.01325×10^5	Pa
$\dot{m}_e(0)$	0	kg/s

Table 3.3: Initial conditions used for the numerical evaluation of Kilicarslan's model equations.

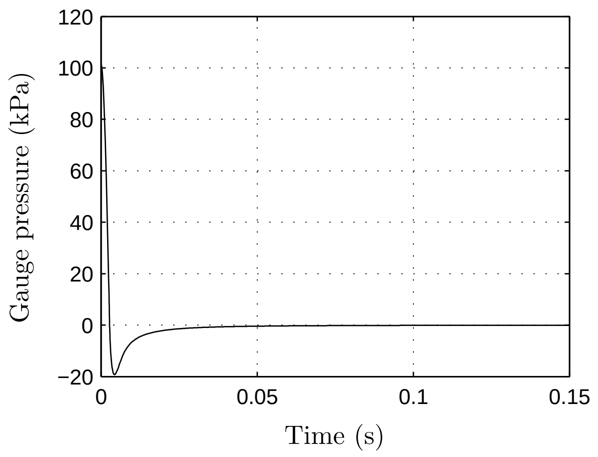


Figure 3.3: Development of the pressure in the combustion chamber over time, as predicted by the model equations of Kilicarslan. The model parameters were given 'reasonable' values; very strong damping results.

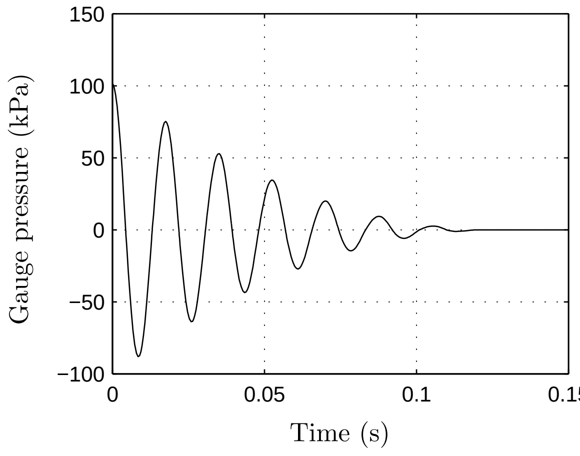


Figure 3.4: Development of the pressure in the combustion chamber over time, as predicted by the model equations of Kilicarslan. Model parameters were changed significantly in order to achieve oscillations. Still, very strong damping remains.

be achieved by decreasing the damping, for example by decreasing the effective flow area of the gas valve, $A_{v,g}$: that decreases k_g, see (3.12), which decreases the constant A, see (3.18), decreasing the damping term in (3.22). If the effective flow area of the air valve is decreased by the same factor, the air-fuel ratio will be unchanged (see (3.14)). An increase of the frequency can be achieved by, for example, increasing the volume of the combustion chamber, or decreasing the tailpipe length, as is suggested by the formula for the frequency of operation (see (3.25)). Only with the use of very different values can some oscillations in the system be achieved. Figure 3.4 shows the results after halving the effective flow area of the gas valve, and taking ten times the combustion chamber volume and a quarter of the tailpipe length used above. Still, the damping is very strong.

It is clear that the model equations cannot correctly describe a stable operating pulse combustor.

3.6 Conclusions

Kilicarslan [17] has suggested a lumped parameter model for a gas-fired Helmholtz pulse combustor with flapper valves. In his paper, an energy balance analysis of the combustion chamber, with simple steady-flow orifice equations describing the reactants inflow and a certain heat release model for the combustion, is linked to a plug flow description of the products in the tailpipe. It leads to a second order differential equation for the pressure in the combustion chamber. Neglecting the damping term, an approximation for the operating frequency of the pulse combustor is arrived at.

In this chapter, the model mentioned above has been examined analytically and numerically. The derivation of the model equations has been given, largely following Kilicarslan's presentation. The heat release model of Kilicarslan comes down to releasing an amount of heat equal to the heat of combustion of the reactants entering the combustion chamber, at the moment of entrance. This can be viewed as burning the entering reactants upon entrance into the combustion chamber. By Rayleigh's criterion, this timing of heat release would dampen oscillations in the system, rather than amplify them. A simple analysis of the model equations shows that pressure fluctuations in the combustion chamber are indeed damped, according to the mathematical model. A numerical investigation of the model equations shows that the damping predicted by the mathematical model is very strong for 'reasonable' choices of the model parameters. It can be concluded that the mathematical model suggested by Kilicarslan cannot describe a stable operating pulse combustor.

The predicted value of the frequency is obtained by neglecting the damping term in the mathematical model, assuming that the damping effect is of negligible effect upon the operating frequency. In the article by Kilicarslan no investigation into the mathematical model is made, except for the predicted frequency, which may explain why the failure of the model to describe a stable pulse combustor has gone unnoticed.

The model may be corrected by modeling the heat release in a different way, allowing for reactants to build up in the combustion chamber and igniting them at a more convenient moment, satisfying Rayleigh's criterion. Since Kilicarslan's model is very similar to the lumped model presented by Ahrens et al. [2], in fact only differing in the way the heat release is modeled, the latter model is worth investigating.

Chapter 4

A model suggested by Ahrens, et al

The mathematical model of Kilicarslan presented in the previous chapter is largely based on that of Ahrens et al. [2], which will be investigated in this chapter. The only real difference is in the modeling of the heat release by combustion. Ahrens et al. assume the reaction zone to be separated in a cool zone occupied by the reactants and a hot zone occupied by the combustion products. Combustion is assumed to take place in a thin flame sheet separating the two zones. The heat release is modeled by the movement of an equivalent plane flame sheet spanning the combustion chamber cross-section, moving at a pressure-independent 'flame-speed'. In this way, if the density of the reactants in the combustion chamber increases, more reactants mass is burnt and more heat is released. At constant temperature of the reactants, an increase in density means an increase in pressure. Thus, with this heat release model, more heat is released at moments of increased pressure, thereby satisfying Rayleigh's criterion.

The assumptions lead to a second-order differential equation, similar to the one given by Kilicarslan, but with a different damping term. According to the mathematical model of Ahrens et al., the heat release indeed amplifies the pressure oscillations. Also, in accordance with Rayleigh's criterion, the entrance of reactants at moments of low pressure attenuates the pressure oscillations. Ahrens et al. assume that stable operation of a pulse combustor can be described by choosing the model parameters (e.g., the constant flame speed), such that these two counteracting effects are in balance. As in the approach of Kilicarslan, the frequency of operation is estimated by neglecting the damping term.

In this chapter, the mathematical model suggested by Ahrens et al. will be investigated, largely following the presentation given in their article. The nomenclature is adjusted in order to ease the comparison of the three models described in this report. In the end, it will be shown that the assumption that stable operation of a pulse combustor can be described by balancing the two counteracting damping effects, is questionable.

4.1 Nomenclature

Roman symbols

A_b	Area of combustion chamber cross-section	$[\,\mathrm{m}^2\,]$
A_{tp}	Area of tailpipe cross-section	$[\,\mathrm{m}^2\,]$
A_v	Effective flow area air/gas valve	$[\,\mathrm{m}^2\,]$
C_D	Discharge coefficient	$[\,-\,]$
c_p	Specific heat for constant pressure	$[\,\mathrm{J/kg\,K}\,]$
c_v	Specific heat for constant volume	$[\,\mathrm{J/kg\,K}\,]$
e	Specific internal energy	$[\,\mathrm{J/kg}\,]$
E	Total energy	$[\,\mathrm{J}\,]$
f	Frequency	$[\,\mathrm{Hz}\,]$
h	Specific enthalpy	$[\,\mathrm{J/kg}\,]$
k_g	Gas valve parameter	$[\,\mathrm{kg}^{1/2}\,\mathrm{m}^{1/2}\,]$
L_{tp}	Tailpipe length	$[\,\mathrm{m}\,]$
$\dot{m}$	Mass flux	$[\,\mathrm{kg/s}\,]$
p	Gauge pressure (in combustion chamber)	$[\,\mathrm{Pa}\,]$
P	Pressure (in combustion chamber)	$[\,\mathrm{Pa}\,]$
$\dot{Q}$	Rate of heat release	$[\,\mathrm{J/s}\,]$
r	Air-fuel mass ratio	$[\,-\,]$
R	Specific gas constant	$[\,\mathrm{J/kg\,K}\,]$
t	Time	$[\,\mathrm{s}\,]$
T	Temperature	$[\,\mathrm{K}\,]$
U_f	Model parameter related to flame speed	$[\,\mathrm{m/s}\,]$
V	Volume	$[\,\mathrm{m}^3\,]$

Greek symbols

ΔH_f	Heat of combustion per unit mass fuel	$[\,\mathrm{J/kg}\,]$
γ	Ratio of specific heats, $= c_p/c_v$	$[\,-\,]$
ω_0	Angular frequency	$[\,\mathrm{rad/s}\,]$
ρ	Density	$[\,\mathrm{kg/m}^3\,]$

Subscripts

0	initial/standard/ambient value
a	air (oxidizer)
b	concerning burning of reactants
cc	combustion chamber
e	exit combustion chamber/entrance tailpipe
g	gas (fuel)
p	products (burnt reactants)
r	reactants (fuel and oxidizer)
tp	tailpipe

4.2 Derivation of model equations

Since Kilicarslan based his mathematical model of a pulse combustor largely on the work by Ahrens et al., the derivations of the model equations are nearly equivalent. The derivation of the model equations of Ahrens et al. will not be

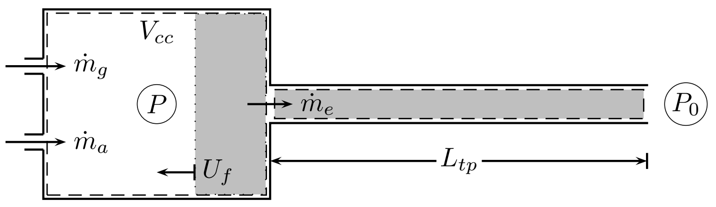


Figure 4.1: Control volume (i.e., the combustion chamber, with volume V_{cc}) of the modeled pulse combustor, with sign conventions and (some) model parameters. The values of the mass fluxes $\dot{m}_g$, $\dot{m}_a$ and $\dot{m}_e$ and the 'flame speed' U_f are taken positive in the directions indicated.

described here in great detail. Rather, only deviations from the derivation of Kilicarslan's model given in the previous chapter, will be highlighted.

Ahrens et al. assume that the burning process inside the combustion chamber takes place in a thin zone (a moving flame front) separating a cold zone containing the reactants, perfectly mixed, from a hot zone containing the combustion products. The state variables are assumed to be uniform in both the zones. The volumes of the two zones add up to the volume of the combustion chamber. Apart from the modeling of the heat release this more clearly defined picture of the combustion process seems to be the only difference with Kilicarslan's presentation. Figure 4.1 shows a schematic of the pulse combustor, with model parameters and sign conventions.

4.2.1 Conservation of energy

The assumptions described in Chapter 3 for the derivation of Kilicarslan's model equations of the gases behaving as perfect gases, of the ratio of specific heats being constant and of the ratio of specific heats for the reactants and products being equal, are also applied here. Taking the combustion chamber as a control volume together with the assumptions on the uniform properties in the two zones lead to the same energy balance equation as (3.8):

$$\frac{dE_{cc}}{dt} = \dot{Q} + h_r \dot{m}_r - h_e \dot{m}_e, \tag{4.1}$$

where E_{cc} is the total energy in the combustion chamber, $\dot{Q}$ is the rate of heat release by combustion, $\dot{m}_r$ and $\dot{m}_e$ are the mass fluxes into and out of the combustion chamber, respectively, and h_r and h_e are the specific (sensible) enthalpies[1] of the reactants and products, respectively. As before (see expression (3.9)), the total energy in the combustion chamber can be written as:

$$E_{cc} = \frac{PV_{cc}}{\gamma - 1}, \tag{4.2}$$

where $\gamma = c_p/c_v$ is the ratio of specific heat for constant pressure, c_p, and for constant volume, c_v, P is the pressure and V_{cc} is the volume of the combustion chamber. The rate of heat release $\dot{Q}$ in the combustion chamber is given by

$$\dot{Q} = \dot{m}_b \frac{\Delta H_f}{1 + r} = \dot{m}_b (h_e - h_r), \tag{4.3}$$

[1]See footnote on page 14

where $\dot{m}_b$ is the mass burning rate of the reactant mixture (to be specified later), ΔH_f is the heat of combustion per unit mass of fuel, r is the air-fuel ratio on mass basis and h_e and h_r are the enthalpies leaving the combustion chamber to the tailpipe and entering the combustion chamber, respectively. Note that Kilicarslan assumed the mass burning rate to be equal to the mass rate of reactants entering the combustion chamber, i.e., $\dot{m}_b = \dot{m}_r$, and thus assumed burning the exact amount of mass entering the chamber at the exact moment of entrance. In Chapter 3 it was shown that then the burning occurs at an inconvenient moment of rarefaction of the gases in the combustion chamber.

4.2.2 Rate of heat release

The heat release is modeled by specifying the burning rate of the reactant mixture. The mass burning rate caused by the flame separating the zones of unburnt (reactants) and burnt gases (products) is assumed to be equivalent to that of a plane flame sheet across a cross-section of the combustion chamber, moving at a certain pressure-independent 'flame speed' U_f relative to the unburnt gases. Denoting the area of the cross-section of the combustion chamber by A_b the mass burning rate is given by

$$\dot{m}_b = \rho_r A_b U_f, \tag{4.4}$$

where ρ_r is the density of the reactants. With T_0 denoting the constant temperature of the reactants and using the perfect gas law there follows

$$\dot{m}_b = \frac{P}{RT_0} A_b U_f = \frac{P_0 + p}{RT_0} A_b U_f, \tag{4.5}$$

where $p = P - P_0$ is the combustion chamber gauge pressure with P_0 the atmospheric pressure and R is the specific gas constant of the reactants. Note that the mass burning rate, and thus the rate of heat release, is directly proportional to the pressure in the combustion chamber. So for this model of the heat release Rayleigh's criterion for amplification of the pressure amplitude is fulfilled at all times.

In their article, Ahrens et al. stress that U_f is not to be regarded as a flame velocity (i.e., the velocity at which the flame front spreads in the combustion chamber), but rather as a system parameter related to the combustion process that attains has a certain value if the pulse combustor is operating stably. Although not explicitly stated, it is clear from the article that U_f is assumed to be constant throughout the derivation of the model equations.

4.2.3 Mass of reactants in combustion chamber

To be able to use the above given model of reactants burning process at all times, it is assumed that there is a certain amount of reactants in the combustion chamber present at all times. The rate of change of the total mass of reactants present in the combustion chamber, M_r, is given by

$$\frac{dM_r}{dt} = \dot{m}_r - \dot{m}_b, \tag{4.6}$$

i.e., by the mass flow rate of reactants entering the combustion chamber minus the mass burning rate of the reactants. In the article of Ahrens et al., this requirement is used only for a stably operating pulse combustor, where there should not be any accumulation or depletion of reactants in the combustion chamber over a cycle. Then the requirement on the mass flow and mass burning rates amounts to

$$\int_{\text{cycle}} (\dot{m}_r - \dot{m}_b)dt = 0, \tag{4.7}$$

where the integration is taken over a period of one cycle.

4.2.4 Mass flow rate into combustion chamber

The modeling of the mass flow rate through the valves as done by Ahrens et al. is already treated above in the derivation of the mass flow rates in Kilicarslan's mathematical model. The result for the mass flow rate of reactants into the reaction chamber is, see (3.13):

$$\dot{m}_r(t) = k_g(1+r)\sqrt{|p|}\,\mathbf{1}_{(-\infty,0)}(p), \tag{4.8}$$

where k_g is a constant parameter corresponding to the flow through the gas valve and $\mathbf{1}_X(x)$ is an indicator function equal to one if x is an element of the set $X \subset \mathbf{R}$ and zero otherwise. The constant k_g is given by, see (3.12):

$$k_g = \sqrt{2\rho_g} C_{D,g} A_{v,g}, \tag{4.9}$$

where ρ_g is the density of the gas, $C_{D,g}$ is a discharge coefficient of the gas valve and $A_{v,g}$ is the effective flow area of the gas valve. The constant air-fuel ratio is given by, see (3.14):

$$r = \sqrt{\frac{\rho_a}{\rho_g}} \frac{C_{D,a} A_{v,a}}{C_{D,g} A_{v,g}}, \tag{4.10}$$

where ρ_a is the density of the air, $C_{D,a}$ is a discharge coefficient of the air valve and $A_{v,a}$ is the effective flow area of the air valve.

4.2.5 Coupling to tailpipe dynamics by conservation of momentum

Modeling of the tailpipe flow is done as shown above in the derivation of Kilicarslan's model. Thus, see (3.15):

$$L_{tp}\frac{d\dot{m}_e}{dt} = pA_{tp}, \tag{4.11}$$

where L_{tp} is the tailpipe length and A_{tp} the area of its cross-section.

4.2.6 Complete set of model equations

The equations above lead to the following system of first-order nonlinear differential equations

$$\frac{dp}{dt} = AF(p) + B(p+P_0) - C\,\dot{m}_e \tag{4.12}$$

$$\frac{d\dot{m}_e}{dt} = Dp \tag{4.13}$$

where

$$A = k_g(1+r)\frac{\gamma-1}{V_{cc}}h_r \tag{4.14}$$

$$B = \frac{A_b U_f}{RT_0}\frac{\gamma-1}{V_{cc}}\frac{\Delta H_f}{1+r} \tag{4.15}$$

$$C = \frac{(\gamma-1)h_e}{V_{cc}} \tag{4.16}$$

$$D = \frac{A_{tp}}{L_{tp}} \tag{4.17}$$

$$F(p) = \sqrt{|p|}\,\mathbf{1}_{(-\infty,0)}(p) \tag{4.18}$$

with (see expression (4.3))

$$h_e = \left(h_r + \frac{\Delta H_f}{1+r}\right). \tag{4.19}$$

By differentiating equation (4.12) and substitution of equation (4.13) a single second-order nonlinear differential equation is derived. As before, the function $F(p)$ is not differentiable in $p = 0$ and its derivative $F'(p)$ will defined as $F'(0) \equiv 0$, i.e., equal to the right-hand derivative of F in $p = 0$. This yields a model equation for the gauge pressure p is the combustion chamber equivalent to that of Ahrens et al.:

$$\frac{d^2p}{dt^2} - (AF'(p) + B)\frac{dp}{dt} + CDp = 0, \tag{4.20}$$

where

$$F'(p) = \frac{dF}{dp} = -\frac{1}{2\sqrt{|p|}}\,\mathbf{1}_{(-\infty,0)}(p). \tag{4.21}$$

To the system of first-order differential equations (4.12)–(4.13), or the second-order differential equation (4.20), appropriate initial conditions must be added, for example by prescribing the initial gauge pressure, $p(0) = p_0$, and the initial mass flux out of the combustion chamber, $\dot{m}_e(0) = \dot{m}_{e,0}$.

The requirement that reactants must be present in the combustion chamber at all times, is presented by the inequality $M_r(t) > 0$ for all t. The total mass of the reactants in the combustion chamber, M_r, satisfies the differential equation (see (4.6))

$$\frac{dM_r}{dt} = \dot{m}_r - \dot{m}_b, \tag{4.22}$$

with $\dot{m}_r$ and $\dot{m}_b$ given by (4.8) and (4.5), respectively. This differential equation should be supplemented with an initial condition, for example by specifying the total mass of reactants $M_{r,0}$ present in the combustion chamber at a certain starting point $t = 0$:

$$M_r(0) = M_{r,0}. \tag{4.23}$$

4.3 Prediction of frequency of pulse combustion

In the article by Ahrens et al., it is noted that the differential equation (4.20) is similar to one describing a linear oscillator with nonlinear damping. The equation can be written as

$$\ddot{x} + h(x, \dot{x}) + \omega_0^2 x = 0, \tag{4.24}$$

where $x = p(t)$, a dot denotes a time derivative, the nonlinear damping term $h(x, \dot{x})$ is defined by

$$h(x, \dot{x}) = -(AF'(x) + B)\dot{x}, \tag{4.25}$$

with $F'(x)$ given by (4.21) and ω_0 is defined as

$$\omega_0 = \sqrt{CD} = \sqrt{(\gamma - 1)h_e}\sqrt{\frac{A_{tp}}{V_{cc}L_{tp}}}. \tag{4.26}$$

It is also noted that the contributions to the damping term by heat release $(-B\dot{x})$ and by reactants mass inflow $(-AF'(x)\dot{x})$ are of opposite sign, since $F'(x) \leq 0$ for all x. Thus, they have counteracting effects on the oscillations, the heat release amplifying the oscillations and the mass inflow of reactants attenuating them.

Ahrens et al. note that for an undamped oscillation the frequency, f_0, would be given by

$$f_0 = \frac{\omega_0}{2\pi} = \frac{\sqrt{(\gamma - 1)h_e}}{2\pi}\sqrt{\frac{A_{tp}}{V_{cc}L_{tp}}}. \tag{4.27}$$

They also mention that “in principle the damping term will cause a frequency shift from the undamped value,” but that in their results this shift was very small. The frequency f_0 can be used as an estimate for the frequency of oscillations of a stable pulse combustion process. They also mention that f_0 is the frequency of a Helmholtz resonator. (Note that $\sqrt{(\gamma - 1)h_e} = \sqrt{(\gamma - 1)c_pT_e} = \sqrt{\gamma RT_e}$, which is the speed of sound in the tailpipe.)

4.4 Searching for a stable oscillation

Ahrens et al. mention that they have solved the model equations numerically and that they found that for a stable oscillating solution the model parameter U_f and the amplitude of the oscillation $p_{\max}$ must both have a specific value for each set of the other model parameters. They proceed by formulating two stability criteria, which are analyzed under simplifying assumptions. The analysis yields two expressions relating U_f and $p_{\max}$, with which the unique values for these parameters can be estimated.

One of the stability criteria is the requirement that there is no accumulation or depletion of reactants in the combustion chamber over a cycle. The other stability criterion is that the pressure amplitude of the oscillations, and the mean pressure, should be independent of time. Ahrens et al. note that in the present case the mean pressure should be zero, as a consequence of the assumption that there is no friction in the tailpipe.

4.4.1 No accumulation/depletion of reactants

The requirement of no accumulation or depletion of reactants is formulated by equation (4.7), repeated here for convenience:

$$\int_{\text{cycle}} (\dot{m}_r - \dot{m}_b)dt = 0, \tag{4.28}$$

with $\dot{m}_b$ and $\dot{m}_r$ given by expressions (4.5) and (4.8), respectively, also repeated here for convenience:

$$\begin{aligned} \dot{m}_r(t) &= k_g(1+r)\sqrt{|p|}\,\mathbf{1}_{(-\infty,0)}(p), && (4.29)\\ \dot{m}_b &= \frac{P_0+p}{RT_0}A_bU_f. && (4.30) \end{aligned}$$

Imposing the simplifying assumption that the pressure variation in time is of the form

$$p(t) = -p_{\max}\cos(\omega_0 t), \tag{4.31}$$

and using the substitution $\tau = \omega_0 t$, Ahrens et al. show that evaluation of the integral in (4.28) leads to

$$2k_g(1+r)\sqrt{p_{\max}}\int_0^{\frac{\pi}{2}}\sqrt{\cos\tau}\,d\tau - 2\pi\frac{P_0}{RT_0}A_bU_f = 0. \tag{4.32}$$

The integral in this equation can be evaluated using complete elliptic integrals K and E, and is given by Ahrens et al. to be

$$I = \int_0^{\frac{\pi}{2}}\sqrt{\cos\tau}\,d\tau = 2\sqrt{2}E\left(\tfrac{1}{2}\sqrt{2}\right) - \sqrt{2}K\left(\tfrac{1}{2}\sqrt{2}\right). \tag{4.33}$$

This gives the relation between $p_{\max}$ and U_f:

$$U_f = \frac{I}{\pi}RT_0\frac{k_g(1+r)}{P_0A_b}\sqrt{p_{\max}}. \tag{4.34}$$

Ahrens et al. evaluate the integral to give $I/\pi = 0.375$ (thus, $I = 1.1781$), which is different from a MATLAB evaluation based on a quadrature algorithm (using the function `quad`) giving $I = 1.1981$ (thus, $I/\pi = 0.381$). Formula (4.33) is correct and can be derived as follows. Using the relation $\cos\tau = 1 - 2\sin^2(\tau/2)$, the integral I can be written as

$$I = \int_0^{\frac{\pi}{2}}\sqrt{\cos\tau}\,d\tau = 2\int_0^{\frac{\pi}{4}}\sqrt{1-2\sin^2\hat{\tau}}\,d\hat{\tau} = 2\,E\left(\frac{\pi}{4},\sqrt{2}\right), \tag{4.35}$$

where $E(\varphi,k) \equiv \int_0^{\varphi}\sqrt{1-k^2\sin^2\phi}\,d\phi$ has been used as definition for the incomplete integral of the second kind, with amplitude φ and modulus k. The modulus in the right-hand side of (4.35) is greater than unity. Most tables on elliptic integrals only deal with moduli less than or equal to unity, as does the MATLAB function `ellipke`[2] for evaluating complete elliptic integrals. A transformation can be applied to express the right-hand side of (4.35) in terms of incomplete integrals of the first and second kind with a modulus less than unity (see [5, 114.01, p. 12]): if $k_1 = 1/k$ and $\sin\varphi_1 = k\sin\varphi$, then

$$E(\varphi,k) = k_1[k^2E(\varphi_1,k_1) + (1-k^2)F(\varphi_1,k_1)], \tag{4.36}$$

[2]Note that the MATLAB function `ellipke` evaluates the complete integrals of the first and second kind for the *parameter* m as argument. The parameter m is related to the *modulus* k by $m = k^2$.

where $F(\varphi, k)$ denotes the incomplete integral of the first kind with amplitude φ and modulus k. It follows that

$$\begin{aligned} I &= 2\,E\left(\frac{\pi}{4}, \sqrt{2}\right) = \sqrt{2}\left[2E\left(\frac{\pi}{2}, \frac{1}{\sqrt{2}}\right) - F\left(\frac{\pi}{2}, \frac{1}{\sqrt{2}}\right)\right] \\ &= 2\sqrt{2}E\left(\tfrac{1}{2}\sqrt{2}\right) - \sqrt{2}K\left(\tfrac{1}{2}\sqrt{2}\right). \end{aligned} \tag{4.37}$$

As a check for the evaluation of the integral I using the MATLAB function `ellipke`, the integral was also evaluated using Table 17.2 in [1, pp. 610–611] for the complete elliptic integrals. Both methods of evaluation yield $I = 1.1981$, consistent with the value previously found by using the MATLAB quadrature function and different from the value given by Ahrens et al. In order to avoid further confusion, the value for the integral I in (4.34) will not be substituted; if needed, the value $I = 1.1981$ can be used as the correct value, or the value $I = 1.1781$ can be used for easy comparison with derived quantities in the article by Ahrens et al.

Note that the value of I in (4.34) comes from the integral of the reactants mass flux $\dot{m}_r$ over a cycle and depends on the pressure variation $p(t)$ as function of time. In general, for $p(t)$ not prescribed, I could be defined as

$$I = \frac{1}{2T_P}\int_{\text{cycle}} \sqrt{|\widehat{p}(t)|}\mathbf{1}_{(-\infty,0)}(\widehat{p}(t))\,dt, \tag{4.38}$$

with $\widehat{p} = p/p_{\max}$ giving the 'shape' of the pressure variation over time, and T_P the period of oscillation. If the second criterion of stability is invoked, i.e., the mean pressure variation over a cycle is zero, the integral over the mass burning rate $\dot{m}_b$ is the same as before and expression (4.34) is valid for I as just given. Thus, the exact relationship between U_f and $p_{\max}$ is given by expression (4.34) for some value of I, which was approximated above by assuming the pressure variation to be sinusoidal with frequency $\frac{\omega_0}{2\pi}$, giving $I = 1.1981$.

4.4.2 Zero mean pressure

In order to arrive at another expression relating U_f and $p_{\max}$, Ahrens et al. use the stability criterion that the mean pressure variation over a cycle must be zero. They use some simplifying assumptions, based on a general shape of the pressure variation as function of time. In Figure 4.2 a sketch of the pressure variation as function of the time is shown for a cycle, with some typical points in time marked on the time axis. The simplifying assumptions, which are not motivated, come down to the following (referring to Figure 4.2):

- the maximum variation in pressure is as great above zero as below zero, i.e., $p_{\max} = -p_{\min}$;
- the ratios of the shaded areas above and below the time axis to the total areas above and below the time axis, respectively, are equal. Thus,

$$\frac{\int_{t_1}^{t_2} p(t)\,dt}{\int_{t_0}^{t_2} p(t)\,dt} = \frac{\int_{t_2}^{t_3} p(t)\,dt}{\int_{t_2}^{t_4} p(t)\,dt}. \tag{4.39}$$

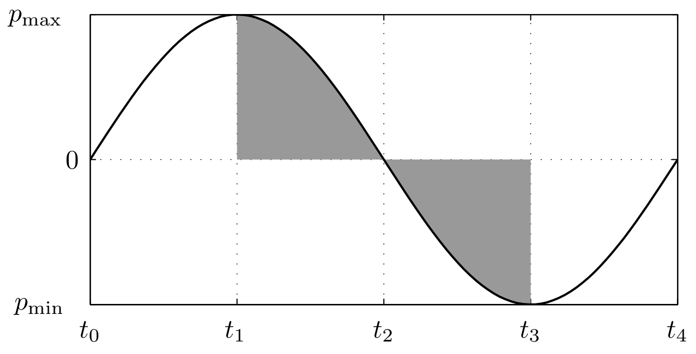


Figure 4.2: Sketch of the gauge pressure as function of the time during a cycle. The marked times on the time axis are: t_0, start of the cycle ($p = 0$); t_1, moment of maximum pressure ($p = p_{\max}$); t_2, moment of zero pressure ($p = 0$); t_3, moment of minimum pressure ($p_{\min} < 0$); and t_4, end of the cycle ($p = 0$).

With the mean pressure variation being zero, the last assumption is equivalent to assuming that the shaded areas are of equal size, with the integrals of opposite sign:

$$\int_{t_1}^{t_2} p(t)\,dt = -\int_{t_2}^{t_3} p(t)\,dt. \tag{4.40}$$

Integration of the second-order nonlinear differential equation (4.20) from time τ_0 to time τ_1 gives

$$\begin{aligned} 0 &= \int_{\tau_0}^{\tau_1} \left(\frac{d^2p}{dt^2} - (AF'(p) + B)\frac{dp}{dt} + CDp \right) dt \\ &= \left[\frac{dp(t)}{dt} - (AF(p(t)) + Bp(t)) \right]_{t=\tau_0}^{\tau_1} + CD \int_{\tau_0}^{\tau_1} p(t)\,dt, \end{aligned} \tag{4.41}$$

where the term with square brackets denotes the difference of the term evaluated for the superscript and the term evaluated for the subscript, $[f(t)]_{t=a}^{b} = f(b) - f(a)$. It follows that

$$\int_{\tau_0}^{\tau_1} p(t)\,dt = -\frac{1}{CD} \left[\frac{dp(t)}{dt} - (AF(p(t)) + Bp(t)) \right]_{t=\tau_0}^{\tau_1}. \tag{4.42}$$

Using the expressions $-p(t_3) = -p_{\min} = p_{\max} = p(t_1)$ and $p(t_2) = 0$, and the function $F(p) = \sqrt{|p|}\mathbf{1}_{(-\infty,0)}(p)$ (see (4.18)), gives

$$AF(p) + Bp = \begin{cases} B\,p_{\max}, & t = t_1 \\ 0, & t = t_2 \\ A\sqrt{p_{\max}} - B\,p_{\max}, & t = t_3 \end{cases} \tag{4.43}$$

With $\frac{dp(t)}{dt} = 0$ for $t = t_1, t_3$, and using expression (4.42), it is easy to show that the assumption (4.40) leads to $B\,p_{\max} = A\sqrt{p_{\max}} - B\,p_{\max}$, or

$$\sqrt{p_{\max}} = \frac{A}{2B} = \frac{k_g(1+r)^2 h_r R T_0}{2A_b U_f \Delta H_f}, \tag{4.44}$$

where the expressions (4.14) and (4.15) for A and B, respectively, have been substituted. (Note that the value of $\frac{dp}{dt}(t = t_2)$ is not needed, since it appears on both sides of the equal sign, and thus drops from the equation.) This is the second relation between $p_{\max}$ and U_f given by Ahrens et al.

Actually, the derivation of expression (4.44) given by Ahrens et al. is somewhat more elaborate than presented above. In the process of deriving this expression, they determine the exact solution of the pressure variation in the interval of the cycle where the pressure is positive. In such an interval, the nonlinear contribution to the damping term caused by the reactants inflow is zero, so that the exact solution in the interval is easily found. From this exact solution, Ahrens et al. infer an additional requirement on the model parameters that need to be fulfilled in order to get a stable operation of the pulse combustor. Some comments on this will be given below in Section 4.5, entitled "Comments on the mathematical model."

4.4.3 Combining the two stability criteria

Substitution of expression (4.34) for U_f into (4.44) leads in a straightforward way to the unique pressure amplitude expressed in the model parameters:

$$p_{\max} = \left(\frac{2I}{\pi}\right)^{-1} \frac{P_0(1+r)h_r}{\Delta H_f}. \tag{4.45}$$

Back-substitution of this expression into (4.34) gives an expression of U_f in other model parameters than $p_{\max}$:

$$U_f = \sqrt{\frac{I}{2\pi}} \frac{RT_0(1+r)^{3/2}}{A_b\sqrt{P_0}} \sqrt{\frac{h_r}{\Delta H_f}}. \tag{4.46}$$

These expressions for the model parameters U_f and $p_{\max}$ are used as estimates for the values that are necessary for a stable oscillation of the pressure in the combustion chamber. More accurate values can be obtained by solving the model equations numerically and requiring the fulfillment of the two stability criteria. Some comments on the imposed requirements to obtain a stable oscillating pressure amplitude are given in the next section.

4.5 Comments on the mathematical model

The process of pulse combustion as modeled by Ahrens et al. is one of combustion inducing pressure increase, resulting in products outflow and a drop in pressure (eventually dropping below ambient pressure), whereupon the cycle starts anew with the inflow of reactants. The heat release as modeled by Ahrens et al. satisfies Rayleigh's criterion at all times and therefore the combustion process continuously amplifies pressure oscillations. The amplitude of the pressure oscillations would grow without limit, if it were not for some limiting factors.

For stable operation of a pulse combustor the model presented by Ahrens et al. requires a balance between damping of pressure oscillations by reactants inflow and amplification thereof by combustion. The model parameters U_f (which is associated with the combustion process and firing rate of the pulse combustor) and $p_{\max}$ (the amplitude of the pressure oscillations) are determined by requiring such a balance.

As mentioned by Ahrens et al., many pulse combustion phenomena are not represented in the model. This raises the question whether the damping by reactants inflow is indeed the most important limiting factor. This question is addressed in this section.

4.5.1 Damping effects

Without the nonlinear damping term $-AF'(p)\frac{dp}{dt}$ in model equation (4.20) caused by the reactants inflow, an analytical solution of the model equation can be to obtained by elementary methods. Then, given initial conditions $p(0) = 0$ and $\frac{dp}{dt}(t = 0) = \dot{p}_0$ for some $\dot{p}_0 > 0$ at a starting point $t = 0$, and provided the solution $p(t)$ of the model equation will return to zero in a finite time span, the solution would be given by

$$p(t) = \frac{\dot{p}_0}{\beta} \mathrm{e}^{\alpha t} \sin(\beta t), \tag{4.47}$$

where

$$\alpha = \frac{B}{2} \tag{4.48}$$

and

$$\beta = \frac{1}{2}\sqrt{|B^2 - 4CD|}. \tag{4.49}$$

Thus, without damping by reactants inflow the pressure amplitude would be amplified each (half-)cycle by the same factor. During the half-cycles that the gauge pressure is positive, there is no reactants inflow and the exact solution of the model equations is given by (4.47). (At least, if the pressure does return to zero after the start of the half-cycle. Conditions for this will be discussed below.) From (4.47) it is clear that the greater (i.e., the more positive) the rate of change of the gauge pressure $\frac{dp}{dt}$ at the beginning of the half-cycle is, the greater is the pressure amplitude during the half-cycle and the more negative is the rate of change of the gauge pressure at the end of the half-cycle.

During the half-cycle that the gauge pressure is negative, the inflow of reactants counteracts the amplification of the pressure amplitude. However, since $F'(p) = -1/(2\sqrt{|p|})$ for $p < 0$ (see (4.21)), the damping effect is less for more negative gauge pressure. At the beginning of reactants inflow (then $p = 0$) the gauge pressure grows negative more rapidly if the rate of change of the gauge pressure $\frac{dp}{dt}$ is more negative, resulting in less damping during the cycle. Thus, if the damping by reactants inflow during a cycle is not sufficient to counteract the amplification of the pressure amplitude by combustion, the rate of change of the gauge pressure at the beginning of the reactants inflow will be more negative in the next cycle, resulting in even less damping. On the other hand, if the damping by reactants inflow is greater than the amplification by combustion, the rate of change of the gauge pressure at the moment of reactants inflow for the next cycle will be less negative than that in the previous cycle, resulting in even more damping. From this it may be expected that either the pressure amplitude will grow without limit (if the damping is not strong enough) or the pressure oscillations will die out (if the damping is too strong). The numerical results in the next section confirm these expectations of the behavior of the solution of the model equations.

From this it may be concluded that, according to the mathematical model of Ahrens et al., the damping by reactants inflow does not drive a pulse combustion system toward stable operation. 'Stable' operation requires an exact relationship between the initial rate of change of the gauge pressure, $\dot{p}_0$ (or, equivalently, the pressure amplitude $p_{\max}$ sought after in Section 4.4), and the model parameter U_f. Any deviation from that relationship and the pressure oscillations will die out or grow without limit. This suggests that other damping factors than reactants inflow are responsible for stabilizing a pulse combustion system. Therefore, the premise that the model parameter U_f (related to the combustion) can be obtained by requiring a balance between the effects of combustion and reactants inflow on the pressure oscillations seems questionable. If other factors are indeed important in limiting pressure oscillations, one might even strive for system parameters that, according to the model of Ahrens et al., would let the pressure oscillations grow without limit. A real pulse combustor designed with such parameters could operate stably due to damping by other factors than reactants inflow.

4.5.2 Requirement of presence of reactants in combustion chamber and burning rate modeling

One requirement for stable operation of a pulse combustor that Ahrens et al. used to determine the model parameter U_f, is that there should be no reactants mass build up or depletion in the combustion chamber. This in fact excludes one physical process that could be a limiting factor on growth of the pressure amplitude in a real pulse combustor: if all the reactants in the combustion chamber are consumed by combustion, there will be no extra heat release, and thus no extra rise in pressure. Ahrens et al. mention that the presence of reactants in the combustion chamber throughout the combustion cycle "appears to be consistent with the observed operation of pulse combustion burners." But this does not mean that the burning model prescribing the burning rate of reactants $\dot{m}_b$ expressed by (4.5) is an appropriate one throughout the cycle. One may expect that when the reactants run low, the reaction rate is no longer representable by the thin flame sheet across the entire combustion chamber cross-section moving at a certain flame speed U_f, but instead might be described by such a flame sheet (using the same flame speed U_f) across a smaller area. Then, as opposed to the burning model of Ahrens et al., reactants could be present throughout the cycle while the heat release diminishes significantly if most of the reactants are consumed by combustion, thereby limiting the amplification of the pressure oscillations. Thus, the requirement in the model of Ahrens et al. that there is always some reactants mixture present in the combustion chamber, combined with the burning model that prescribes the consumption of the mixture at a certain rate only dependent on the pressure in the combustion chamber, seems to impose a limit on the capability of the model to describe stable operating pulse combustors.

4.5.3 Conditions for oscillatory solutions

In the analysis on the damping effects, it was assumed that the solution $p(t)$ of the model equations starting at some point $t = 0$ in time, with initial conditions $p(0) = 0$ and $\frac{dp}{dt}(t = 0) = \dot{p}_0 > 0$, would return to zero in a finite time span. With the nonlinear damping term in model equation (4.20) being zero if $p > 0$, it is easy to show that for the gauge pressure to return to zero the relation $B^2 - 4CD < 0$, or equivalently $B/\sqrt{CD} < 2$, must be satisfied: if this condition is satisfied, the solution of the model equations will be oscillatory on time intervals of positive gauge pressure; otherwise, it will be continuously increasing in time. Formulated in model parameters, the condition is

$$\frac{A_b U_f \Delta H_f}{R T_0 (1 + r)} \sqrt{\frac{(\gamma - 1) L_{tp}}{h_e A_{tp} V_{cc}}} < 2 \tag{4.50}$$

(see (4.15), (4.16) and (4.17) for expressions of B, C and D in model parameters). Similar arguments[3] lead Ahrens et al. to suspect that stable pulse combustor operation may be impossible for model parameters not satisfying inequality (4.50). They mention that in practice this condition is usually met, so that it is not a serious restriction in the design of pulse combustors. They also mention that the condition could not be put to the test in their article, since for all the pulse combustors they used data of, the system parameters satisfied the condition.

According to the model of Ahrens et al., sets of system parameters not satisfying inequality (4.50) result in a continuously increasing pressure in the combustion chamber. This would ultimately result in the consumption of all the reactants present in the combustion chamber, which violates a criterion of Ahrens et al. for stable pulse combustion burner operation. Considering the discussion above on the requirement of presence of reactant mixture throughout a cycle in combination with the burning rate model used by Ahrens et al., it seems questionable to consider inequality (4.50) as a restriction on design choices of the system parameters for pulse combustors. For a better understanding of the conditions under which a pulse combustion burner may operate stably, additional physical processes should be included in the model and/or physical processes should be modeled in greater detail (for example the burning rate).

4.6 Numerical implementation and experiments

The system of model equations (4.12)–(4.19) is numerically implemented using MATLAB [30] software. The standard MATLAB ODE-solver `ode45`[4] is used to solve the equations. The implementation in MATLAB is straightforward; the codes used can be found in Appendix B.

In order to solve the model equations numerically, the model parameters and the initial conditions need to be specified. The model parameters chosen by Ahrens et al. that are related to the dimensions of a pulse combustor are

[3]They use the exact solution of the model equations for positive gauge pressure, but state incorrectly that the condition $B/\sqrt{CD} < 2$ must hold, for otherwise the gauge pressure "would approach zero asymptotically and, therefore, cannot switch over to the negative pressure regime."

[4]See footnote 2 on page 18

Parameter	Value used	Unit
A_b	8.11×10^{-3}	m^2
A_{tp}	7.92×10^{-4}	m^2
$A_{v,g}$	6.26×10^{-5}	m^2
L_{tp}	1.52	m
r	19.8	–
V_{cc}	1.97×10^{-3}	m^3

Table 4.1: Model parameters associated with pulse combustor dimensions as used for numerical evaluation of the model equations. Values are taken from Table 1 (AGA experimental burner No. 2) in the article [2] of Ahrens et al.

based on existing pulse combustion burners. In their article, the dimensional properties of the burners are presented clearly in Table 1 on AGA Experimental Burners. Of these, the dimensions associated with burner No. 2 are used for the numerical evaluation of the model equations in this report. This choice is made, because of the available data presented in the table this particular burner operates with the gas pressure supplied at the ambient (atmospheric) value, which is an assumption used in developing the model equations. Other listed pulse combustion burners operate with a slightly pressurized gas supply. There is one other burner listed operating with the gas pressure supplied at the ambient value: it has the same dimensions except for a tailpipe that is more than twice as long. The burner with the shorter tailpipe is listed operating with a mean chamber pressure that is less than that with the burner with the longer tailpipe ($7,421\,\mathrm{Pa}$ versus $11,204\,\mathrm{Pa}$). Since no friction was assumed in the derivation of the model equations, the mean chamber pressure should be zero in the mathematical model. The influence of friction in the tailpipe is less with the burner with the shorter tailpipe, thus that tailpipe length is chosen to numerically evaluate the model equations. The values chosen for the model parameters related to the dimensions of the pulse combustor are listed in Table 4.1.

The other model parameters that are needed for the numerical evaluation of the model equations, and some auxiliary parameters used to compute the model parameters, are listed in Table 4.2. As in the article of Ahrens et al., the values chosen for the model parameters are based on methane as fuel and air property values. The parameter values needed are computed or looked up in literature (see the caption of the table). For the ambient temperature and associated air property values $T_0 = 300$ K is used, almost equal to the value of $T_0 = 530°$ R used by Ahrens et al. Since the appropriate values for the discharge coefficients $C_{D,g}$ of the gas valves of the pulse combustion burners were unknown to Ahrens et al., they chose (rather arbitrarily) the value $C_{D,g} = 0.6$ for it.

The initial conditions needed for numerical evaluation of the model equations (4.12)–(4.19) are chosen in the following way. The numerical evaluation is chosen to start at a point in time ($t = 0$) of maximum gauge pressure p_0, thus $\frac{dp}{dt}(t = 0) = 0$. Substitution of $\frac{dp}{dt}(t = 0)$ and $p(0) = p_0$ in differential equation (4.12) for

Parameter	Value used	Unit	Comment
$C_{D,g}$	0.6	–	
c_p	1009.7	J/kg K	air prop., [28, p.346]
h_e	$= h_r + \frac{\Delta H_f}{1+r}$	J/kg	
h_r	$= c_p T_0$	J/kg	
k_g	$= \sqrt{2\rho_g} C_{D,g} A_{v,g}$	$\mathrm{kg}^{1/2}\,\mathrm{m}^{1/2}$	
MW_g	16.043	kg/kmol	mol. weight, [28, p.343]
P_0	1.01325×10^5	Pa	$= 1\,\mathrm{atm}$
R_a	287	J/kg K	air prop., [28, p.343]
R_g	$= R_u/\mathrm{MW}_g$	J/kg K	
R_u	8.31441×10^3	J/kmol K	univ. gas const., [28, in back]
T_0	300	K	
ΔH_f	50.016×10^6	J/kg	LHV [33, p.543]
ρ_g	$= P_0/(R_g T_0)$	$\mathrm{kg/m}^3$	

Table 4.2: Model parameters (and auxiliary values to compute model parameters) used for the numerical evaluation of the model equations of Ahrens et al. The value of the parameter $C_{D,g}$ is that chosen by Ahrens et al. [2]; the other parameter values are based on air property values (as used by Ahrens et al.) at a reactants temperature of 300 K (close to the choice of 530° R of Ahrens et al.) and the fuel choice of methane. The reference temperature and pressure for the LHV of methane is 298.15 K and 1 atm, respectively. The air and fuel properties were looked up in tables in [28] and [33].

the rate of change of the gauge pressure yields the following initial conditions:

$$p(0) = p_0, \tag{4.51}$$

$$\dot{m}_e(0) = \dot{m}_{e,0} = \frac{B}{C}(p_0 + P_0), \tag{4.52}$$

where $\dot{m}_{e,0}$ denotes the prescribed initial mass flux into the pulse combustor tailpipe.

The initial (maximum) gauge pressure should be related to the model parameter U_f in some way in order to obtain a 'stable' oscillating solution. As a first guess, the values of p_0 and U_f are chosen as those given by $p_{\max}$ and U_f in expressions (4.45) and (4.46), respectively, that resulted from the analysis of Ahrens et al. The value of I in these expression is chosen to be consistent with the (incorrect) value used by Ahrens et al., i.e. $I/\pi = 0.375$, so that $p_{\max}$ and U_f are given by

$$p_{\max} = \frac{1}{0.75}\frac{P_0(1+r)h_r}{\Delta H_f}, \tag{4.53}$$

$$U_f = \sqrt{0.1875}\frac{RT_0(1+r)^{3/2}}{A_b\sqrt{P_0}}\sqrt{\frac{h_r}{\Delta H_f}}. \tag{4.54}$$

This first choice of parameters p_0 ($= p_{\max}$) and U_f does not result in a stable oscillating solution: the initial oscillation quickly dies out. Keeping U_f and $p_{\max}$ constant at the values chosen above, the initial gauge pressure p_0 is adjusted

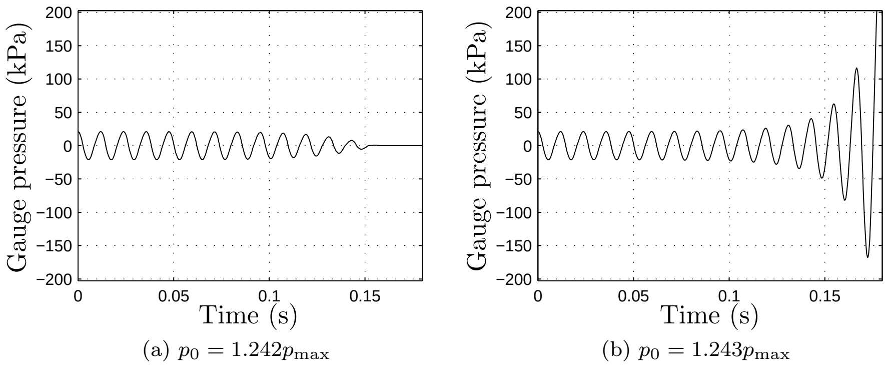


Figure 4.3: Gauge pressure as function of time for different, but very close, initial conditions $p_0 = (1+\delta)p_{\max}$: (a) $\delta = 0.242$, (b) $\delta = 0.243$.

(increased if the oscillations die out, decreased if the amplitude of oscillations grows continuously) by choosing it to be a multiple of $p_{\max}$:

$$p_0 = (1+\delta)p_{\max}, \tag{4.55}$$

for a certain $\delta > -1$. This procedure for searching a 'stable' oscillating solution will yield a set of parameters p_0 and U_f satisfying only the requirement of zero mean gauge pressure in the combustion chamber, not the requirement of the presence of reactants in the combustion chamber throughout the cycle. However, for illustrating that the sought-after stable oscillating solution does not exist (i.e., the pressure oscillations will either grow without limit, or they will die out), it will suffice. The solution of the model equations will show a similar behavior for the pair of model parameters p_0 and U_f for which the requirement on the reactants appears to be satisfied. Of course, the fulfillment of this requirement cannot be maintained if the pressure oscillations grow too large, or die away.

Figure 4.3 shows plots of the computed gauge pressure versus time for the values $\delta = 0.242$ and $\delta = 0.243$. The difference of the initial gauge pressures is very small and the computed gauge pressures appear to be oscillating stably in the beginning. However, after some time has passed, the graphs show significantly different behavior of the gauge pressure: for $\delta = 0.242$ the oscillations die out, while for $\delta = 0.243$ the amplitude increases without bounds.

The difference in evolution of the computed gauge pressure for the two choices of δ is illustrated by the phase diagram shown in Figure 4.4. The phase path associated with $\delta = 0.242$ starts at $(0, 1.242p_{\max})$ and spirals inwardly to the origin. The phase path associated with $\delta = 0.243$ starts at $(0, 1.243p_{\max})$ and spirals outwardly.

The graphs clearly shows the influence of the damping: around $p = 0$, where for $p < 0$ the damping is the greatest, the rate of change of the gauge pressure in absolute sense is decreased and change in gauge pressure is severely inhibited. The graphs suggest the existence of a limit cycle for a value of δ somewhere between 0.242 and 0.243. However, for a value of δ slightly different the damping will continuously increase or decrease and while for some time the phase paths may stay close to the limit cycle, they will eventually diverge.

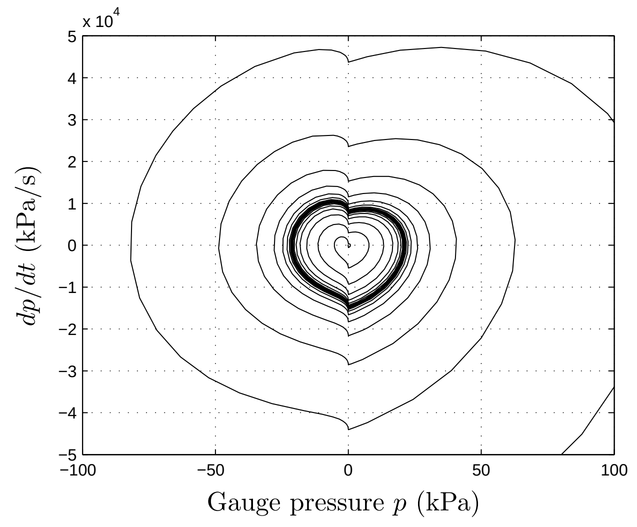


Figure 4.4: Trajectories in phase plane corresponding to the nonlinear model equation (4.20) for two different, but very close, initial conditions. One trajectory, corresponding to an initial condition $p_0 = 1.242p_{\max}$, spirals inwardly toward the origin. The other trajectory, corresponding to the initial condition $p_0 = 1.243p_{\max}$, spirals outwardly.

4.7 Conclusions

In their article [2] Ahrens et al. present a lumped parameter model for a pulse combustion burner. An energy balance analysis of the combustion chamber, with simple steady-flow orifice equations describing the reactants inflow and a heat release given by combustion of the reactants at a constant volumetric rate, is linked to a plug flow description of the products in the tailpipe. No energy losses, for example by friction in the tailpipe or heat transfer to the combustion chamber wall, are modeled. The analysis of Ahrens et al. leads to a second-order differential equation for the gauge pressure in the combustion chamber with a negative (linear) damping term associated with the combustion process and a positive (nonlinear) damping term associated with the inflow of reactants. The constant volumetric burning rate is equivalent to that of a thin flame sheet across the combustion chamber moving at a certain constant 'flame speed'. Ahrens et al. found this constant 'flame speed' and the amplitude of the pressure variation in the combustion chamber to be uniquely determined if the solution of their model equation is to satisfy two criteria for stable pulse combustion burner operation. One criterion they mention is that the mean gauge pressure in the combustion chamber should be zero (since no energy losses are modeled). The other criterion is that there should be no accumulation or depletion of reactants in the combustion chamber. Also, the model equations of Ahrens et al. impose a certain condition on the model parameters which must be satisfied for the solution to be oscillatory. Ahrens et al. conclude from this that a pulse combustion burner with design parameters not satisfying that condition might not work properly.

In this chapter, the mathematical model presented by Ahrens et al. has been examined analytically and numerically. The derivation of the model equations has been given, largely following the presentation of Ahrens et al. The analysis shows that the impact of damping of the pressure oscillation by inflow of reactants diminishes if the amplitude of the oscillation increases. According to the model equations, the inflow of reactants causes pressure oscillations either to die out or to continuously increase in amplitude. Other physical processes than the inflow of reactants, for example a burning rate that is different from the one used in the model, seem to be responsible for reaching a stable oscillation. The premise that model parameters of a stably operating pulse combustor should satisfy the conditions that yield a 'stable' oscillatory solution of the model equations, is therefore questionable. Besides the inflow of reactants, other physical processes that may influence the stability of oscillations should be included in the model in order to gain more insight in the conditions under which a pulse combustor may operate properly.

Chapter 5

A model suggested by Richards, et al

Note: In a Bachelor's thesis project at the Eindhoven University of Technology, Van Mullem and Bastiaans [34] examined the model of Richards et al. [26], which is discussed in this chapter. They gave a quick investigation into the effect of changing the air/fuel ratio. This report was studied, but the discussion in this chapter focuses only on the work presented by Richards et al.

Richards et al. [26] introduced a mathematical model to describe a combustion system with a continuous fuel supply. This is different from a typical pulse combustor, where fuel periodically enters the combustion chamber because of time-dependent pressure differences over valves. They show that pulsation can occur even for a continuous fuel supply. To indicate the impact of heat transfer on the pulsating process, they termed it *thermal pulse combustion*.

The model is a lumped parameter model, taking the combustion chamber as a control volume. The amount of energy in the combustion chamber is changed by inflow of reactants, combustion, outflow of combustion products, and heat transfer to the chamber wall. The combustion process is modeled by a one-step Ahrrenius law for a bimolecular reaction between fuel and oxidizer. The gases in the combustion chamber are assumed to be well-stirred. The tailpipe flow is modeled as a slug flow, i.e. of constant density and driven by the pressure difference over the tailpipe. Flow from the combustion chamber into the tailpipe is assumed to be isentropic. Wall friction of the tailpipe gases is taken into account. It is assumed that the gases are perfect, and that all mixtures of reactants and products have the same (constant) specific heats. By applying conservation of mass, energy and species to the control volume, and coupling this to the tailpipe dynamics by conservation of momentum, Richards et al. derived a system of four ordinary differential equations.

The numerical results obtained by Richards et al. showed three possible modes of operation, depending on the model parameters: steady combustion, flame extinction, or oscillatory combustion. The influence of microscale mixing was also investigated numerically. The numerical results showed a qualitative agreement to experimental data.

In this chapter, the mathematical model suggested by Richards et al. will be investigated. The nomenclature used by Richards et al. is adjusted in order to ease the comparison with the other models that are investigated in this report. The numerical results of Richards et al. will be reproduced (for as far as this was possible). As will be seen in the discussion of the results in this report, the phase difference between heat release and the acoustic oscillations provides an intuitive explanation of the system's behavior. Although the importance of the phase difference may be obvious from Rayleigh's criterion, Richards et al. did not give it much attention in the discussion of their results.

5.1 Nomenclature

Roman symbols

A–G	Auxiliary constants	
$\widehat{A}$	Kinetic constant for fuel reaction	$[\,\mathrm{m^3/kg\,s\,K^{\frac{1}{2}}}\,]$
A'	Modified kinetic constant, $= A\rho_0\sqrt{T_0}S_r$	$[\,1/\mathrm{s}\,]$
A_{tp}	Area of tailpipe cross-section	$[\,\mathrm{m^2}\,]$
A_s	Surface area of combustion zone	$[\,\mathrm{m^2}\,]$
c_p	Specific heat for constant pressure	$[\,\mathrm{J/kg\,K}\,]$
c_v	Specific heat for constant volume	$[\,\mathrm{J/kg\,K}\,]$
D_{tp}	Tailpipe diameter	$[\,\mathrm{m^2}\,]$
e	Specific internal energy	$[\,\mathrm{J/kg}\,]$
E_A	Activation energy	$[\,\mathrm{J/mol}\,]$
f	Friction coefficient (in tailpipe)	$[\,-\,]$
F_f	Friction force (in tailpipe)	$[\,\mathrm{N}\,]$
h	Specific enthalpy	$[\,\mathrm{J/kg}\,]$
$\widehat{h}$	Heat transfer coefficient	$[\,\mathrm{W/m^2\,K}\,]$
L_1	First characteristic length, $= V_{cc}/A_s$	$[\,\mathrm{m}\,]$
L_2	Second characteristic length, $= V_{cc}/A_{tp}$	$[\,\mathrm{m}\,]$
L_{tp}	Tailpipe length	$[\,\mathrm{m}\,]$
M	Mach number	$[\,-\,]$
$\dot{m}$	Mass flow rate	$[\,\mathrm{kg/s}\,]$
$\mathrm{MW_X}$	Molecular weight of chemical compound X	$[\,\mathrm{g/mol}\,]$
$\mathbf{n}$	Outward normal on control volume boundary	$[\,-\,]$
P	Pressure (in combustion chamber)	$[\,\mathrm{Pa}\,]$
$\mathbf{q}$	Heat flow per unit of area	$[\,\mathrm{J/m^2\,s}\,]$
$\dot{Q}$	Rate of heat release per unit volume	$[\,\mathrm{J/m^3\,s}\,]$
R	Specific gas constant	$[\,\mathrm{J/kg\,K}\,]$
$\dot{R}_f$	Fuel reaction rate	$[\,\mathrm{kg/m^3\,s}\,]$
$\dot{R}_o$	Oxygen reaction rate	$[\,\mathrm{kg/m^3\,s}\,]$
R_u	Universal gas constant	$[\,\mathrm{J/mol\,K}\,]$
RR	Auxiliary reaction rate	$[\,1/\mathrm{s}\,]$
$\widehat{RR}$	Auxiliary reaction rate, modified for mixing time	$[\,1/\mathrm{s}\,]$
S_r	Stoichiometric oxygen-fuel mass ratio	$[\,-\,]$
t	Time	$[\,\mathrm{s}\,]$

Continued on next page

Roman symbols, continued

T	Temperature	[K]
$\widetilde{T}_{\text{act}}$	Dimensionless activation temperature, $= E_A/R_uT_0$	[–]
T_w	Wall temperature of combustion chamber	[K]
u	Velocity of tailpipe plug flow	[m/s]
u_c	Characteristic tailpipe velocity, $= \dot{m}_i/\rho_0 A_{tp}$	[m/s]
$\mathbf{v}$	Velocity vector (in tailpipe)	[m/s]
v^2	$= \mathbf{v}\cdot\mathbf{v}$	[$\mathrm{m}^2/\mathrm{s}^2$]
V	Volume	[m^3]
Y	Mass fraction (in combustion chamber)	[–]
Z	Auxiliary parameter, $= \dot{m}/V$	[$\mathrm{kg/m^3\,s}$]

Greek symbols

γ	Ratio of specific heats, $= c_p/c_v$	[–]
ΔH_f	Heat of combustion per unit mass fuel	[J/kg]
Ω	Control volume	[–]
ρ	Density	[$\mathrm{kg/m^3}$]
τ_c	Combustion time, $= c_p\rho_0T_0/\dot{Q}$	[s]
τ_{cm}	Combined combustion and mixing time, $= \tau_c + \tau_m$	[s]
τ_f	Flow time, $= \rho_0/Z_i$	[s]
τ_{HT}	Heat transfer time, $= L_1c_p\rho_0T_0/\widehat{h}T_w$	[s]
τ_m	Mixing time	[s]
τ_{wall}	Friction shear stress at tailpipe wall	[$\mathrm{N/m^2}$]
χ_{X}	molal fraction of chemical compound X	[–]

Other symbols

$\tilde{}$	(Over variable) Indicates normalized variable
$\partial\Omega$	Boundary of control volume

Subscripts

0	initial/standard/ambient value
cc	combustion chamber
e	exit combustion chamber/entrance tailpipe
f	fuel (except for F_f and τ_f, see above)
i	inlet of combustion chamber
mix	mixture
o	oxygen
w	combustion chamber wall

Superscripts

$*$	average value
tot	total (or, stagnation) variables

5.2 Derivation of model equations

In this section the derivation of the model equations will be given, largely following the presentation of Richards et al. For clarity, some in-between steps in

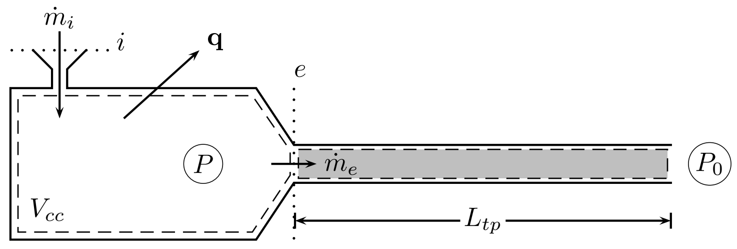


Figure 5.1: Control volume (i.e., the combustion chamber, with volume V_{cc}) of the modeled thermal pulse combustor, with sign conventions and (some) model parameters. The values of the mass fluxes $\dot{m}_i$ at the inlet i, and $\dot{m}_e$ at the tailpipe entrance e are taken positive in the directions indicated. The heat flux $\mathbf{q}$ is directed outward of the combustion chamber.

the derivation are added where Richards et al. only show the resulting equations or mention the way to get them.

5.2.1 Conservation of energy

The combustion chamber is chosen as a control volume. Let Ω denote the control volume and $\partial\Omega$ its boundary. See Figure 5.1 for a schematic of the control volume and sign conventions. Conservation of energy over this control volume is expressed by

$$\frac{d}{dt}\int_{\Omega}\rho(e+\frac{1}{2}v^2)\,dV = -\int_{\partial\Omega}\rho\mathbf{v}\cdot\mathbf{n}(e+\frac{1}{2}v^2)\,dA - \int_{\partial\Omega}P\mathbf{v}\cdot\mathbf{n}\,dA + \int_{\Omega}\dot{Q}\,dV - \int_{\partial\Omega}\mathbf{q}\cdot\mathbf{n}\,dA \quad (5.1)$$

where ρ is the density, e is the specific internal energy, P is the pressure, $\dot{Q}$ is the rate of heat release per unit volume, $\mathbf{v}$ is the fluid velocity, v^2 denotes $\mathbf{v}\cdot\mathbf{v}$, $\mathbf{q}$ is the heat flow per unit area out of the control volume, and $\mathbf{n}$ is the outwardly directed normal on the boundary. The integral on the left-hand side represents the rate of change of the total amount of energy in the control volume. The first integral on the right-hand side is the energy flux through the boundary out of the control volume. The second integral on the right-hand side is the rate of flow work done by the control volume. The third integral on the right-hand side is the rate of heat release inside the control volume and the last integral is the heat flux through the boundary out of the control volume.

It is assumed that the specific kinetic energy inside the control volume is negligibly small compared to the specific internal energy. The gases (reactants and the products) are assumed to be perfect, i.e., they obey the perfect gas law

$$P = \rho RT, \quad (5.2)$$

with R the gas constant and T the temperature, and the ratio of specifics heats is constant:

$$\gamma = \frac{c_p}{c_v} = \text{constant}, \quad (5.3)$$

where c_p is the specific heat for constant pressure and c_v is the specific heat for constant volume. Also, the specific heats and gas constants of the reactants are assumed to be equal to those of the products. The control volume is considered to be well-mixed and therefore the state variables are considered to have a uniform value over the volume. With the specific internal energy given by

$$e = c_v T, \tag{5.4}$$

and with the relation

$$R = c_p - c_v, \tag{5.5}$$

and under the assumptions mentioned above, the left-hand side of equation (5.1) can be written as

$$\frac{d}{dt}\int_{\Omega} \rho(e + \frac{1}{2}v^2)\, dV = V_{cc}\frac{1}{\gamma - 1}\frac{dP}{dt}, \tag{5.6}$$

where V_{cc} is the volume of the combustion chamber.

The conditions at the inlet and exit of the combustion chamber are assumed to arise from isentropic acceleration of the gases to their inlet and outlet velocities. Thus,

$$\left(\frac{T^{\text{tot}}}{T}\right)_{i/e} = \left(1 + \frac{\gamma-1}{2}M^2\right)_{i/e} = \left(\frac{P^{\text{tot}}}{P}\right)_{i/e}^{\frac{\gamma-1}{\gamma}} = \left(\frac{\rho^{\text{tot}}}{\rho}\right)_{i/e}^{\gamma-1}, \tag{5.7}$$

with the subscript i/e denoting the variables at either the inlet (i) or exit (e) of the combustion chamber. Here T^{tot}, P^{tot} and ρ^{tot} are the total, or stagnation, temperature, pressure and density, respectively. (See Thompson [31, pp. 267–268] for a derivation of these equations.) Because inside the combustion chamber $v^2/2$ is set to zero (it is assumed to be negligibly small compared to the specific internal energy), the state variables inside the combustion chamber are equal to their total state variables:

$$T_{cc}^{\text{tot}} = T, \quad P_{cc}^{\text{tot}} = P, \quad \rho_{cc}^{\text{tot}} = \rho, \tag{5.8}$$

where the subscript cc expresses the correspondence to the state variables inside the control volume (i.e., inside the combustion chamber). The Mach number at the inlet is assumed to be negligibly small, $M_i \approx 0$. For the total temperatures at the inlet and exit there follows

$$\begin{aligned} c_p T_{i/e} + \frac{v_{i/e}^2}{2} &= c_p\left(1 + \frac{1}{2c_p}M_{i/e}^2\gamma R\right)T_{i/e} \\ &= c_p\left(1 + \frac{\gamma-1}{2}M_{i/e}^2\right)T_{i/e} = c_p T_{i/e}^{\text{tot}}. \end{aligned} \tag{5.9}$$

Thus, using the relation $R = c_p - c_v$ (5.5) and with $T_i^{\text{tot}} = T_i$ and $T_e^{\text{tot}} = T$,

$$\begin{aligned} \int_{\partial\Omega} \mathbf{v}\cdot\mathbf{n}\left(\rho(e + \frac{v^2}{2}) + P\right) dA &= \int_{\partial\Omega} \rho\mathbf{v}\cdot\mathbf{n}\left(\frac{v^2}{2} + e + RT\right) dA \\ &= \int_{\partial\Omega} \rho\mathbf{v}\cdot\mathbf{n}\left(c_p T + \frac{v^2}{2}\right) dA = c_p(\dot{m}_e T - \dot{m}_i T_i), \end{aligned} \tag{5.10}$$

where use is made of the identities $(\rho\mathbf{v}\cdot\mathbf{n})_i = -\dot{m}_i$ and $(\rho\mathbf{v}\cdot\mathbf{n})_e = \dot{m}_e$, the subscripts indicating values at inlet and exit and $\dot{m}_e$ being the mass flow rate at the entrance of the tailpipe. Note that this result could have been obtained easier by using the specific enthalpy $h = e + P/\rho = c_pT$ and using the fact that an isentropic flow is adiabatic (i.e., no change in heat content) so that for a perfect gas the enthalpy does not change.

It should be stressed that in the derivation of equation (5.10), use of the relation $T_e^{\text{tot}} = T$, prescribing the total temperature at the tailpipe entrance by isentropic acceleration of gases from the combustion chamber, is valid only for flow exiting the combustion chamber. For flow in the reverse direction, from tailpipe into the combustion chamber, the total temperature at the tailpipe entrance only *contributes* to the temperature in the combustion chamber, and should be independent from it. This means that for flow into the combustion chamber (i.e. when $\dot{m}_e < 0$), the term $c_p\dot{m}_eT$ in equation (5.10) should be replaced by $c_p\dot{m}_eT_e^{\text{tot}}$ for some appropriate T_e^{tot} based on the temperature and velocity of the tailpipe gases. Although Richards et al. do not explicitly describe how they implemented the isentropic acceleration/deceleration of the tailpipe gases, the term $c_p\dot{m}_eT$ appears in their formulas, without making any distinction between the cases $\dot{m}_e < 0$ and $\dot{m}_e \geq 0$. It introduces a modeling error. Most of the numerical results show a negative mass flux at the tailpipe entrance (i.e., flow into the combustion chamber) only for a small portion of a cycle, and at a relatively low value in absolute sense. Therefore the effect of the modeling error may be small, although this should be confirmed more rigorously. It would be preferable to correct the model equations to account for flow from the tailpipe back into the combustion chamber, but since this chapter is focused on evaluation of the model and numerical results presented by Richards et al., this will not be pursued here. For the remainder of the derivation of the model equations, it will be assumed that the effect of flow from the tailpipe back into the combustion chamber is negligible.

The heat flux per unit of area at the boundary of the control volume given by Newton's cooling law is $\mathbf{q}\cdot\mathbf{n} = \widehat{h}(T - T_w)$, where T_w is the combustion chamber wall temperature and $\widehat{h}$ is a heat transfer coefficient. With $\widehat{h}$ assumed to be constant and the assumption of uniformity inside the control volume, the last two integrals of equation (5.1) are given by

$$\int_\Omega \dot{Q}\,dV = \dot{Q}V_{cc}, \tag{5.11}$$

$$\int_{\partial\Omega} \mathbf{q}\cdot\mathbf{n}\,dA = -\widehat{h}(T_w - T)A_s, \tag{5.12}$$

where A_s is the surface area of the combustion zone.

For convenience Richards et al. introduce the following definitions:

$$Z_i = \frac{\dot{m}_i}{V_{cc}}, \tag{5.13}$$

$$Z_e = \frac{\dot{m}_e}{V_{cc}}, \tag{5.14}$$

$$L_1 = \frac{V_{cc}}{A_s}, \tag{5.15}$$

They call the parameter L_1 the first characteristic length and announce a second characteristic length to be introduced later. It will be associated with a characteristic velocity in the tailpipe.

Substituting the expressions for the integrals (5.6), (5.10)–(5.12) in the energy balance equation (5.1) and using the definitions (5.13)–(5.15), the energy balance equation (5.1) can be written as

$$\frac{1}{\gamma-1}\frac{dP}{dt} = c_p(T_i Z_i - T Z_e) + \dot{Q} + \frac{\widehat{h}}{L_1}(T_w - T). \tag{5.16}$$

5.2.2 Conservation of mass

Under the assumption of uniformity, conservation of mass over the control volume gives $d(\rho V_{cc})/dt = \dot{m}_i - \dot{m}_e$. With the definitions (5.13)–(5.14) this yields

$$\frac{d\rho}{dt} = Z_i - Z_e. \tag{5.17}$$

5.2.3 Introducing dimensionless state variables and characteristic times

The dimensionless state variables $\widetilde{P}$, $\widetilde{T}$ and $\widetilde{\rho}$ are introduced by

$$\widetilde{P} = \frac{P}{P_0}, \tag{5.18}$$

$$\widetilde{T} = \frac{T}{T_0}, \tag{5.19}$$

$$\widetilde{\rho} = \frac{\rho}{\rho_0}, \tag{5.20}$$

where P_0 is the ambient pressure, T_0 is the ambient temperature and $\rho_0 = P_0/RT_0$ (these constants are characteristic state parameters). Using these relations in the energy conservation equation (5.16) with $T_i = T_0$ and $P_0 = \rho_0 T_0 R$, and noting that $\frac{P_0}{\gamma-1} = \frac{c_p}{\gamma}\rho_0 T_0$ yields

$$\frac{d\widetilde{P}}{dt} = \gamma\left(\frac{Z_i}{\rho_0} + \frac{\dot{Q}}{c_p\rho_0 T_0} + \frac{\widehat{h}T_w}{L_1 c_p \rho_0 T_0}\right) - \gamma\left(\frac{Z_e}{\rho_0} + \frac{\widehat{h}}{L_1 c_p \rho_0}\right)\widetilde{T}. \tag{5.21}$$

For the mass conservation equation (5.17) it is easy to see that

$$\frac{d\widetilde{\rho}}{dt} = \frac{Z_i}{\rho_0} - \frac{Z_e}{\rho_0}. \tag{5.22}$$

At this point Richards et al. introduce the following 'characteristic times' that are recognizable in the equations (5.21)–(5.22):

$$\tau_f = \frac{\rho_0}{Z_i}, \tag{5.23}$$

$$\tau_{HT} = \frac{L_1 c_p \rho_0 T_0}{\widehat{h}T_w} \tag{5.24}$$

$$\tau_c = \frac{c_p \rho_0 T_0}{\dot{Q}}. \tag{5.25}$$

They call τ_f a flow time, indicating the time that it takes non-reacting isothermal gases to pass through the combustion chamber. It can be compared to the cold residence time of a steady-state reactor, but since in the pulsating case it is not a residence time, the term ‘flow time’ was introduced for τ_f. The characteristic time τ_{HT} is associated with the heat transfer and τ_c is associated with the combustion process. Note that τ_c is not a constant because of $\dot{Q}$ in the denominator (to be modeled later). Since a characterictic parameter is usually a constant, it may be confusing to call τ_c a characteristic time. Also, it will be shown later (see p. 67) that the ‘characteristic time’ τ_{HT} is not characteristic for the heat transfer process, contrary to what the terminology may suggest. Nonetheless, in this report the terminology of Richards et al. is followed, and the constant τ_{HT} and variable τ_c will be called characteristic times.

Using the characteristic times in the energy and mass conservation equations (5.21)–(5.22) gives

$$\frac{d\widetilde{P}}{dt} = \gamma\left(\frac{1}{\tau_f}+\frac{1}{\tau_c}+\frac{1}{\tau_{HT}}\right) - \gamma\left(\frac{Z_e}{\rho_0}+\frac{1}{\tau_{HT}}\frac{T_0}{T_w}\right)\widetilde{T} \tag{5.26}$$

and

$$\frac{d\widetilde{\rho}}{dt} = \frac{1}{\tau_f} - \frac{Z_e}{\rho_0}. \tag{5.27}$$

Richards et al. use a differential equation for the temperature rather than the mass conservation equation (5.27). It is derived as follows. Using the characteristic state parameters (5.18)–(5.20) and the perfect gas law gives $\widetilde{P} = \widetilde{\rho}\widetilde{T}$ and using the mass conservation equation (5.27) it follows that

$$\frac{d\widetilde{P}}{dt} = \frac{d}{dt}(\widetilde{\rho}\widetilde{T}) = \widetilde{T}\frac{d\widetilde{\rho}}{dt} + \widetilde{\rho}\frac{d\widetilde{T}}{dt} = \widetilde{T}\left(\frac{1}{\tau_f}-\frac{Z_e}{\rho_0}\right) + \frac{\widetilde{P}}{\widetilde{T}}\frac{d\widetilde{T}}{dt}. \tag{5.28}$$

Solving for $\frac{d\widetilde{T}}{dt}$ gives

$$\frac{d\widetilde{T}}{dt} = \frac{\widetilde{T}}{\widetilde{P}}\frac{d\widetilde{P}}{dt} - \left(\frac{1}{\tau_f}-\frac{Z_e}{\rho_0}\right)\frac{\widetilde{T}^2}{\widetilde{P}}. \tag{5.29}$$

With substitution of $\frac{d\tilde{P}}{dt}$ from equation (5.26) the time derivative of the dimensionless temperature is expressed in terms of the dimensionless pressure and temperature:

$$\frac{d\widetilde{T}}{dt} = \gamma\left(\frac{1}{\tau_f}+\frac{1}{\tau_c}+\frac{1}{\tau_{HT}}\right)\frac{\widetilde{T}}{\widetilde{P}} - \left((\gamma-1)\frac{Z_e}{\rho_0}+\frac{1}{\tau_f}+\frac{\gamma}{\tau_{HT}}\frac{T_0}{T_w}\right)\frac{\widetilde{T}^2}{\widetilde{P}}. \tag{5.30}$$

5.2.4 Rate of heat release and mass fractions in combustion chamber

The rate of heat release (per unit of volume), $\dot{Q}$, is simply the heat of reaction per unit of fuel mass, ΔH_f, times the fuel mass reaction rate (per unit of volume), $\dot{R}_f$, i.e.

$$\dot{Q} = \Delta H_f \dot{R}_f. \tag{5.31}$$

The mass fuel rate of reaction is modeled by a bimolecular reaction rate law between fuel and oxygen:

$$\dot{R}_f = \widehat{A}T^{1/2}\rho^2 Y_o Y_f \exp(-E_A/R_u T). \tag{5.32}$$

where $\widehat{A}$ is some kinetic constant, Y_f and Y_o are the mass fractions of fuel and oxygen, respectively, E_A is the activation energy and R_u is the universal gas constant. The mass oxygen rate of reaction is simply the stoichiometric oxygen-fuel mass ratio, S_r, times the mass fuel rate of reaction:

$$\dot{R}_o = S_r \dot{R}_f. \tag{5.33}$$

Relations for the mass fractions Y_f and Y_o can be derived by considering mass conservation of species in the control volume under the assumption of uniformity inside the control volume. Mass conservation of species for the fuel, for example, is expressed by

$$\frac{d}{dt}\int_{\Omega} \rho Y_f dV = -\int_{\partial\Omega} \rho Y_f \mathbf{v}\cdot\mathbf{n} dA - \int_{\Omega} \dot{R}_f dV, \tag{5.34}$$

where the left-hand side is the amount of fuel mass added to the control volume per unit of time, the first integral on the right-hand side is the fuel mass flux out of the control volume and the second integral on the right-hand side is the amount of fuel mass consumed by combustion in the control volume per unit of time. Under the assumption of uniformity inside the control volume this leads to

$$\frac{d}{dt}(\rho Y_f) = -(Y_{f,e}Z_e - Y_{f,i}Z_i) - \dot{R}_f, \tag{5.35}$$

where $Y_{f,i}$ and $Y_{f,e}$ are the fuel mass fractions at the inlet and exit of the combustion chamber, respectively. Applying the chain rule of differentiation yields

$$\rho\frac{dY_f}{dt} + Y_f\frac{d\rho}{dt} = -(Y_{f,e}Z_e - Y_{f,i}Z_i) - \dot{R}_f. \tag{5.36}$$

This equation can be expressed in the dimensionless pressure and temperature by using the identities $\rho = \rho_0\widetilde{\rho}$ and $\widetilde{\rho} = \widetilde{P}/\widetilde{T}$ and substitution of the mass conservation equation (5.27) for $\frac{d\widetilde{\rho}}{dt}$:

$$\frac{\widetilde{P}}{\widetilde{T}}\frac{dY_f}{dt} = -\left(\frac{1}{\tau_f} - \frac{Z_e}{\rho_0}\right)Y_f - (Y_{f,e}\frac{Z_e}{\rho_0} - Y_{f,i}\frac{Z_i}{\rho_0}) - \frac{\dot{R}_f}{\rho_0}. \tag{5.37}$$

From (5.31) and (5.25) it follows that

$$\frac{\dot{R}_f}{\rho_0} = \frac{\dot{Q}}{\rho_0\Delta H_f} = \frac{1}{\tau_c}\frac{c_pT_0}{\Delta H_f} \tag{5.38}$$

and with $\frac{Z_i}{\rho_0} = \frac{1}{\tau_f}$ by definition of τ_f (see (5.23)) equation (5.37) can be written as

$$\frac{\widetilde{P}}{\widetilde{T}}\frac{dY_f}{dt} = -\left(Y_{f,e}\frac{Z_e}{\rho_0} - Y_{f,i}\frac{1}{\tau_f}\right) - \left(\frac{1}{\tau_f} - \frac{Z_e}{\rho_0}\right)Y_f - \frac{1}{\tau_c}\frac{c_pT_0}{\Delta H_f}. \tag{5.39}$$

With the assumption of uniformity inside the control volume it follows that $Y_{f,e} = Y_f$, thus

$$\frac{dY_f}{dt} = \frac{\widetilde{T}}{\widetilde{P}}\left(\frac{1}{\tau_f}(Y_{f,i} - Y_f) - \frac{1}{\tau_c}\frac{c_pT_0}{\Delta H_f}\right). \tag{5.40}$$

Note that the relation $Y_{f,e} = Y_f$ used in the derivation of equation (5.40) is valid only for flow from the combustion chamber into the tailpipe. For flow in the reverse direction, from the tailpipe into the combustion chamber, the fuel present in the tailpipe only contributes to the mass fuel fraction in the combustion chamber, and should be independent from it. Equation (5.40) is used as model equation by Richards et al. without making any distinction between $Z_e \geq 0$ (i.e., flow is from the combustion chamber into the tailpipe) and $Z_e < 0$ (i.e., flow is from the tailpipe into the combustion chamber). Recall that the model equations of this chapter are derived under the assumption that the flow from the tailpipe back into the combustion chamber is negligible (see the discussion in the first new paragraph after equation (5.10) on page 48).

The equation for conservation of the oxygen mass fraction can be derived in a completely analogous manner, leading to a similar expression only differing in the last term because the reaction rate is a factor S_r higher (see (5.33) and (5.38)):

$$\frac{dY_o}{dt} = \frac{\widetilde{T}}{\widetilde{P}} \left(\frac{1}{\tau_f}(Y_{o,i} - Y_f) - \frac{1}{\tau_c} \frac{S_r c_p T_0}{\Delta H_f} \right). \tag{5.41}$$

An expression for the characteristic combustion time τ_c in terms of the mass fractions and the dimensionless pressure $\widetilde{P} = P/P_0$ and temperature $\widetilde{T} = T/T_0$ is given by

$$\begin{aligned} \frac{1}{\tau_c} &\overset{(5.25)}{=} \frac{\dot{Q}}{c_p \rho_0 T_0} \overset{(5.31)}{=} \frac{\dot{R}_f \Delta H_f}{c_p \rho_0 T_0} \\ &\overset{(5.32)}{=} \frac{\Delta H_f}{c_p T_0} \rho_0 T_0^{1/2} \widehat{A} \frac{\widetilde{P}^2}{\widetilde{T}^{3/2}} Y_o Y_f \exp\left(-\frac{E_A}{R_u T_0 \widetilde{T}} \right). \end{aligned} \tag{5.42}$$

Richards et al. simplified the equations by assuming stoichiometric inlet conditions and complete combustion, i.e., burnt fuel is completely converted to carbon dioxide and water molecules. Then, at all times the oxygen mass fraction is given by $Y_o = S_r Y_f$. Substituting this expression for Y_o in (5.42) and writing $A' = \widehat{A} \rho_0 T_0^{1/2} S_r$ and $\widetilde{T}_{\text{act}} = E_A / R_u T_0$ gives

$$\frac{1}{\tau_c} = A' \frac{\Delta H_f}{c_p T_0} \frac{\widetilde{P}^2}{\widetilde{T}^{3/2}} Y_f^2 \exp(-\widetilde{T}_{\text{act}}/\widetilde{T}). \tag{5.43}$$

Values used by Richards et al. for propane reaction were obtained from a paper by Kretschmer and Odgers [18] and are given by $A' = 3.85 \times 10^8 \, \text{s}^{-1}$ and $\widetilde{T}_{\text{act}} = 50$, where the value $\widetilde{T}_{\text{act}}$ corresponds to an activation energy of 30 kcal/kmol.

Richards et al. remark that this reaction rate can be used for fuel lean combustion, although the computed oxygen levels will be artificially low in that case. But this seems to be inconsistent with the general reaction rate that is described in the paper of Kretschmer and Odgers [18], on which Richards et al. based their reaction rate parameters. The general reaction rate that is given in the article of Kretschmer and Odgers depends on the equivalence ratio, which is absent in the reaction rate that Richards et al. prescribed. How the reaction rate of Richards et al. is distilled from the one given by Kretschmer and Odgers should be examined more closely.

5.2.5 Coupling to tailpipe dynamics by conservation of momentum

For the coupling of gases exiting the combustion chamber with the dynamics in the tailpipe it is assumed that the gases in the tailpipe move as a slug flow, i.e., moving as a whole under the pressure difference between the entrance of the tailpipe and the exit of the tailpipe (at atmospheric pressure). A certain wall friction force F_f is assumed to be acting in the tailpipe. Applying Newton's law then gives

$$(P_e - P_0)A_{tp} + F_f = \int_{\Omega_{tp}} \rho \, dV \, \frac{du}{dt}, \tag{5.44}$$

where A_{tp} is the tailpipe cross-section area, u is the velocity of the tailpipe gases (taken positive if directed toward the tailpipe exit), and the integral is taken over the tailpipe volume. Richards et al. approximate the volume integral of the density in the tailpipe by

$$\int_{\Omega_{tp}} \rho \, dV = \rho_e A_{tp} L_{tp}, \tag{5.45}$$

where L_{tp} is the tailpipe length (i.e., $A_{tp}L_{tp}$ is the tailpipe volume), yields

$$(P_e - P_0)A_{tp} + F_f = \rho_e A_{tp} L_{tp} \frac{du}{dt}. \tag{5.46}$$

Note that Richards et al. assume that the (average) density in the tailpipe at any moment is equal to the density of the gases entering the tailpipe at that moment. This means that at moments of low density at the tailpipe entrance, the mass of the gases in the tailpipe is underestimated, while it is overestimated when the density is high. Therefore an error in the velocity of the gases in the tailpipe is produced. The impact of the error introduced in the velocity is not clear. The fact that the velocity appears not only in the momentum equation, but, via the mass outflow ($\dot{m}_e$), also in the model equations for the pressure and temperature, obscures it even further.

The issue can easily be resolved by introducing an extra differential equation, which describes the rate of change of the average density in the tailpipe. Using $\rho_{tp} = \frac{m_{tp}}{V_{tp}}$ for the average density in the tailpipe, where m_{tp} denotes the mass of the tailpipe gases and $V_{tp} = L_{tp}A_{tp}$ is the tailpipe volume, and assuming that the tailpipe gases are of uniform density and flow toward the tailpipe exit, it follows that

$$\frac{d\rho_{tp}}{dt} = \frac{1}{V_{tp}} \lim_{\Delta t \to 0} \frac{\Delta m_{tp}}{\Delta t} = \frac{(\rho_e - \rho_{tp})A_{tp}u}{V_{tp}} = \frac{(\rho_e - \rho_{tp})u}{L_{tp}}. \tag{5.47}$$

Instead of the density ρ_e at the tailpipe entrance, the average density of the tailpipe gases ρ_{tp} should be used in the derivation of the momentum equation.

Preliminary results incorporating the adjustments to the tailpipe density as described above, suggest that the modifications may have a significant effect. For example, under the default parameter values introduced later in this chapter, adjusting the momentum equation as described above decreases the peak-to-peak pressure amplitude of oscillations in the combustion system from 71 kPa to 47 kPa, and increases the frequency from 150 Hz to 164 Hz. However, since this chapter is focused on evaluation of the model and numerical results of Richards et al., the differential equation for the average density in the tailpipe and associated changes to the model equations are not implemented here.

A characteristic velocity u_c and an associated second characteristic length L_2 are defined by

$$u_c \equiv \frac{\dot{m}_i}{\rho_0 A_{tp}} = \frac{Z_i}{\rho_0}\frac{V_{cc}}{A_{tp}} = \frac{1}{\tau_f}\frac{V_{cc}}{A_{tp}} \tag{5.48}$$

and

$$L_2 \equiv \frac{V_{cc}}{A_{tp}}. \tag{5.49}$$

Thus $L_2 = u_c\tau_f$, expressing the characteristic length L_2 in terms of the characteristic velocity u_c and the flow time τ_f. Introducing the dimensionless velocity $\widetilde{u} = u/u_c$ and the dimensionless state variables $\widetilde{P}_e = P_e/P_0$, $\widetilde{T}_e = T_e/T_0$, $\widetilde{\rho}_e = \rho_e/\rho_0$, equation (5.46) can be written as

$$P_0(\widetilde{P}_e - 1)A_{tp} + F_f = A_{tp}L_{tp}\rho_0\widetilde{\rho}_e\frac{L_2}{\tau_f}\frac{d\widetilde{u}}{dt} \tag{5.50}$$

and thus

$$\frac{d\widetilde{u}}{dt} = (\widetilde{P}_e - 1)\frac{P_0\tau_f}{L_{tp}L_2\rho_0\widetilde{\rho}_e} + \frac{\tau_f}{L_2}\frac{F_f}{A_{tp}L_{tp}\rho_0\widetilde{\rho}_e}. \tag{5.51}$$

The friction force in the tailpipe is given by the product of the surface area of the tailpipe wall and the friction shear stress at the wall. Taking the tailpipe to be cylindrical, its surface area is given by $\pi D_{tp}L_{tp}$, where D_{tp} is the tailpipe diameter. Using Schlichting's [27, pp. 596–597] friction shear stress on the tailpipe wall $\tau_{\text{wall}} = \frac{1}{8}f\rho_e u^2$, where f is the dimensionless coefficient of resistance (here called the friction coefficient), the wall friction force can be expressed as

$$F_f = -\frac{1}{8}(\pi D_{tp}\,L_{tp})f\rho_e u^2\frac{u}{|u|} = -\frac{\pi}{8}D_{tp}\,L_{tp}\,f\rho_0\widetilde{\rho}_e\left(\frac{L_2}{\tau_f}\right)^2\widetilde{u}^2\frac{\widetilde{u}}{|\widetilde{u}|}, \tag{5.52}$$

where the factor $-u/|u|$ gives the right direction of the friction force, i.e., against the flow direction. Substitution of this expression for the friction force into equation (5.51) and using the relations $P_e = \rho_e RT_e$, $P_0 = \rho_0 RT_0$ (thus $\widetilde{\rho}_e = \widetilde{P}_e/\widetilde{T}_e$) and $A_{tp} = \frac{\pi}{4}D_{tp}^2$ gives

$$\frac{d\widetilde{u}}{dt} = (\widetilde{P}_e - 1)\frac{RT_0\tau_f}{L_{tp}L_2}\frac{\widetilde{T}_e}{\widetilde{P}_e} - \frac{L_2 f}{2D_{tp}\tau_f}\frac{\widetilde{u}^3}{|\widetilde{u}|}. \tag{5.53}$$

5.2.6 Complete set of model equations

As mentioned before, the state variables in the tailpipe entrance are obtained by isentropic acceleration of the gases leaving the combustion chamber to the tailpipe entrance Mach number. For completeness, the equations relating the pressures and temperatures in the tailpipe and the combustion chamber are worked out below. Richards et al. do not explicitly describe how the state variables at the tailpipe entrance are obtained, except that isentropic acceleration/deceleration is used.

By using the total state relations (5.7), the state variables in the tailpipe can be expressed in their total (or, stagnation) variables, which, for flow from the combustion chamber into the tailpipe, are equal to the state variables in the combustion chamber. Using the Mach number $M_e = u/\sqrt{\gamma R T_e}$ for the tailpipe flow, this yields

$$\begin{aligned} \frac{T}{T_e} &= 1 + \frac{\gamma-1}{2} M_e^2 = 1 + \frac{\gamma-1}{2} \frac{u^2}{\gamma R T_e}, && (5.54) \\ \frac{P}{P_e} &= \left(\frac{T}{T_e}\right)^{\frac{\gamma}{\gamma-1}}. && (5.55) \end{aligned}$$

Solving equation (5.54) for T_e gives

$$T_e = T - \frac{\gamma-1}{2} \frac{u^2}{\gamma R} = T\left(1 - \frac{\gamma-1}{2} \frac{u^2}{\gamma R T}\right). \quad (5.56)$$

Thus, the temperature and pressure at the tailpipe entrance, in normalized form, are given by

$$\begin{aligned} \widetilde{T}_e &= \widetilde{T}\left(1 - \frac{\gamma-1}{2}\left(\frac{L_2}{\tau_f}\right)^2 \frac{\widetilde{u}^2}{\gamma R T_0 \widetilde{T}}\right), && (5.57) \\ \widetilde{P}_e &= \widetilde{P}\left(\frac{\widetilde{T}_e}{\widetilde{T}}\right)^{\frac{\gamma}{\gamma-1}}. && (5.58) \end{aligned}$$

Note that, because the state variables at the tailpipe entrance are derived from those in the combustion chamber (via isentropic acceleration), these equations are valid only for flow from the combustion chamber into the tailpipe.

The last term that needs to be specified to complete the mathematical model is Z_e/ρ_0. This term is specified by considering mass conservation: with $\dot{m}_e = A_{tp}\rho_e u$ it follows that

$$\frac{\dot{m}_e}{\rho_0} = A_{tp} u_c \widetilde{\rho}_e \widetilde{u} = A_{tp} \frac{L_2}{\tau_f} \widetilde{\rho}_e \widetilde{u} \quad (5.59)$$

and thus, by definitions of Z_e and L_2 and since $\widetilde{\rho}_e = \widetilde{P}_e/\widetilde{T}_e$,

$$\frac{Z_e}{\rho_0} = \frac{\dot{m}_e}{\rho_0 V_{cc}} = \frac{A_{tp}}{V_{cc}} \frac{L_2}{\tau_f} \widetilde{\rho}_e \widetilde{u} = \frac{1}{\tau_f} \frac{\widetilde{P}_e}{\widetilde{T}_e} \widetilde{u}, \quad (5.60)$$

completing the model equations.

The model equations are summarized by

$$\begin{aligned} \frac{d\widetilde{P}}{dt} &= \gamma\left(\frac{1}{\tau_f} + \frac{1}{\tau_c} + \frac{1}{\tau_{HT}}\right) - \gamma\left(\frac{Z_e}{\rho_0} + \frac{1}{\tau_{HT}} \frac{T_0}{T_w}\right)\widetilde{T}, && (5.61) \\ \frac{d\widetilde{T}}{dt} &= \gamma\left(\frac{1}{\tau_f} + \frac{1}{\tau_c} + \frac{1}{\tau_{HT}}\right)\frac{\widetilde{T}}{\widetilde{P}} \\ &\quad - \left((\gamma-1)\frac{Z_e}{\rho_0} + \frac{1}{\tau_f} + \frac{\gamma}{\tau_{HT}} \frac{T_0}{T_w}\right)\frac{\widetilde{T}^2}{\widetilde{P}}, && (5.62) \\ \frac{d\widetilde{u}}{dt} &= (\widetilde{P}_e - 1)\frac{R T_0 \tau_f}{L_{tp} L_2} \frac{\widetilde{T}_e}{\widetilde{P}_e} - \frac{L_2 f}{2 D_{tp} \tau_f} \frac{\widetilde{u}^3}{|\widetilde{u}|}, && (5.63) \\ \frac{dY_f}{dt} &= \frac{\widetilde{T}}{\widetilde{P}}\left(\frac{1}{\tau_f}(Y_{f,i} - Y_f) - \frac{1}{\tau_c} \frac{c_p T_0}{\Delta H_f}\right). && (5.64) \end{aligned}$$

The time-dependent variables at the tailpipe entrance (found from isentropic acceleration of the gases in the combustion chamber) and the time-dependent combustion time τ_c are given by

$$\widetilde{T}_e = \widetilde{T}\left(1 - \frac{\gamma-1}{2}\left(\frac{L_2}{\tau_f}\right)^2 \frac{\widetilde{u}^2}{\gamma R T_0 \widetilde{T}}\right), \tag{5.65}$$

$$\widetilde{P}_e = \widetilde{P}\left(\frac{\widetilde{T}_e}{\widetilde{T}}\right)^{\frac{\gamma}{\gamma-1}}, \tag{5.66}$$

$$\frac{Z_e}{\rho_0} = \frac{1}{\tau_f}\frac{\widetilde{P}_e}{\widetilde{T}_e}\widetilde{u}, \tag{5.67}$$

$$\frac{1}{\tau_c} = A'\,\frac{\Delta H_f}{c_p T_0}\,\frac{\widetilde{P}^2}{\widetilde{T}^{3/2}}\,Y_f^2 \exp(-\widetilde{T}_{\text{act}}/\widetilde{T}). \tag{5.68}$$

The constant parameters are given by

$$L_1 = \frac{V_{cc}}{A_s}, \tag{5.69}$$

$$L_2 = \frac{V_{cc}}{A_{tp}}, \tag{5.70}$$

$$\tau_f = \frac{\rho_0}{Z_i} = \frac{\rho_0 V_{cc}}{\dot{m}_i}, \tag{5.71}$$

$$\tau_{HT} = \frac{c_p L_1 \rho_0 T_0}{\widehat{h} T_w}. \tag{5.72}$$

5.3 Numerical implementation

In this and the following section, the system of model equations (5.61)–(5.72) is numerically investigated. This section focuses on the numerical implementation of the model equations. Numerical results will be discussed in the next section. First the system of model equations (5.61)–(5.72) is rewritten to a form more suitable for numerical implementation, as will be explained below.

5.3.1 Rewriting the model equations

Close inspection of the system of equations (5.61)–(5.72) reveals that numerical implementation of several parameters needs to be done carefully. It concerns the 'characteristic times': τ_f, τ_c and τ_{HT}. Of these parameters, only the inverse of τ_c and τ_{HT} are used, while τ_f shows up in the equations both inverted and not inverted. For the sake of numerical experimenting, it is useful if in the numerical implementation of the model equations the heat transfer to the combustion chamber wall can be 'turned off' (for example by specifying $\widehat{h} = 0$) or that the combustion process can be 'turned off' (for example by specifying $\Delta H_f = 0$). A glance at expression (5.72) for τ_{HT} (with $\widehat{h}$ in its denominator) makes it clear that this parameter should be implemented numerically by specifying and using only its inverse. The same goes for τ_c (see expression (5.68)), although the factor $\frac{1}{\tau_c}$ in the term $\frac{1}{\tau_c}\frac{c_p T_0}{\Delta H_f}$ in equation (5.64) should be substituted by the expression (5.68), yielding $A'\frac{\widetilde{P}^2}{\widetilde{T}^{3/2}}Y_f^2\exp(-\widetilde{T}_{\text{act}}/\widetilde{T})$ for the term in equation

(5.64). Implementing τ_f so that its inverse $\frac{1}{\tau_f} = \frac{\dot{m}_i}{\rho_0 V_{cc}}$ may take the value 0 is rather complicated (if not impossible) because both τ_f (in (5.63)) and $\frac{1}{\tau_f}$ (in (5.61)–(5.65) and in (5.67)) are used. It is useful to be able to freely specify the mass influx $\dot{m}_i$, and thus $1/\tau_f$, including the value 0 because then the model equations can be used to describe a pulse combustor with varying mass inflow (e.g., in the case of a combustor equipped with flapper valves). The problems with the parameter τ_f arise from the non-dimensionalizing of the velocity of the fluid in the tailpipe with the factor $u_c = L_2/\tau_f$ (see expressions (5.48) and (5.49)). It is clear that in case of a pulse combustor with flapper valves, for example, the mass inflow and the fluid velocity in the tailpipe are hardly directly related. A more natural factor for non-dimensionalizing the fluid velocity in the tailpipe would be the speed of sound of the ambient environment $\sqrt{\gamma R T_0}$, with which the speed in the tailpipe (and thus the particle velocity in the tailpipe) can be compared. Retracing the steps leading to the system of model equations (5.61)–(5.72) shows that all the parameters τ_f in equations (5.61), (5.62) and (5.64) are related to the mass influx $\dot{m}_i$, while all those in equation (5.63) and expressions (5.65) and (5.67) are related to the tailpipe fluid velocity u_c. Thus, the problem with τ_f can be solved by introducing a different characteristic velocity in cases where τ_f is related to (the inverse of) the tailpipe velocity and numerically implementing and using only the inverse of τ_f where τ_f is related to (the inverse of) the mass influx. Then the value $\dot{m}_i = 0$ can be used (yielding $\frac{1}{\tau_f} = 0$).

The system of model equations (5.61)–(5.72) will now be rewritten, taking into account the comments mentioned above. A characteristic speed for the fluid velocity in the tailpipe is denoted by u_0 in order to avoid confusion with the u_c used earlier. The following substitutions are made in the model equations (5.61)–(5.72). The parameter $1/\tau_f$ is replaced by the constant A in equations where it is associated with the mass influx, thus in model equations (5.61), (5.62) and (5.64); it is replaced by u_0/L_2 where it is associated with the tailpipe fluid velocity, thus in (5.63), (5.65) and (5.67). The parameter $1/\tau_c$ (see (5.68)) is written as $1/\tau_c = B \cdot RR$, with $B = \frac{\Delta H_f}{c_p T_0}$ and RR the reaction rate per unit of mass $RR = \frac{\dot{R}_f}{\rho_0} = A' \frac{\widetilde{P}^2}{\widetilde{T}^{3/2}} Y_f^2 \exp(-\widetilde{T}_{\text{act}}/\widetilde{T})$ (see (5.38)). To be able to vary the wall temperature T_w independently of the heat transfer coefficient $\widehat{h}$ the parameter $\frac{1}{\tau_{HT}}$ is replaced by $\frac{1}{\tau_{HT}} = CD$ with $C = \frac{\widehat{h}}{c_p L_1 \rho_0}$ and $D = \frac{T_w}{T_0}$. For ease of notation three extra constants are introduced. Two are given by the constants in the two terms of the equation (5.63) for the tailpipe fluid velocity: $E = \frac{RT_0}{L_{tp} u_0}$ and $F = \frac{f u_0}{2 D_{tp}}$. (Note that the substitution $u_0 = \frac{L_2}{\tau_f}$ is applied.) The term $\frac{\widetilde{u}^3}{|\widetilde{u}|}$ in the same equation is replaced with the equivalent $\widetilde{u}|\widetilde{u}|$, thereby avoiding divisions by zero. The third extra constant, G, is given by defining $\widetilde{Z}_e \equiv \frac{\widetilde{P}_e}{\widetilde{T}_e}\widetilde{u}$ (cf. expression (5.67), coming from (5.60)) and requiring that $G\widetilde{Z}_e = \frac{Z_e}{\rho_0}$. Using expression (5.60) this gives $G = \frac{u_0}{L_2}$. The differential equation for the temperature (5.62) is replaced by the equivalent equation (5.29) from which the final form was found. Since $\frac{d\widetilde{P}}{dt}$ must be computed anyway, this saves some computations and is more compact. Note that the equation for the temperature (5.62) could also be replaced by the much simpler equation for the density (5.27), and then use the perfect gas law to obtain the temperature. In order to be able to make a direct comparison of the results obtained here with those obtained by Richards et al., this will not be implemented.

The adjusted equations are:

$$\frac{d\widetilde{P}}{dt} = \gamma\left(A + B\cdot RR + CD - (C + G\widetilde{Z}_e)\widetilde{T}\right), \tag{5.73}$$

$$\frac{d\widetilde{T}}{dt} = \left(\frac{d\widetilde{P}}{dt} - (A - G\widetilde{Z}_e)\widetilde{T}\right)\frac{\widetilde{T}}{\widetilde{P}}, \tag{5.74}$$

$$\frac{d\widetilde{u}}{dt} = E(\widetilde{P}_e - 1)\frac{\widetilde{T}_e}{\widetilde{P}_e} - F\,\widetilde{u}\,|\widetilde{u}|, \tag{5.75}$$

$$\frac{dY_f}{dt} = (A(Y_{f,i} - Y_f) - RR)\,\frac{\widetilde{T}}{\widetilde{P}}, \tag{5.76}$$

where RR, $\widetilde{P}_e$, $\widetilde{T}_e$ and $\widetilde{Z}_e$ are functions of $\widetilde{P}$, $\widetilde{T}$, $\widetilde{u}$ and Y_f given by

$$RR = A'\,\frac{\widetilde{P}^2}{\widetilde{T}^{3/2}}\,Y_f^2\exp(-\widetilde{T}_{\text{act}}/\widetilde{T}), \tag{5.77}$$

$$\widetilde{T}_e = \widetilde{T}\left(1 - \frac{\gamma-1}{2}\frac{u_0^2}{\gamma R T_0}\frac{\widetilde{u}^2}{\widetilde{T}}\right), \tag{5.78}$$

$$\widetilde{P}_e = \widetilde{P}\left(\frac{\widetilde{T}_e}{\widetilde{T}}\right)^{\frac{\gamma}{\gamma-1}}, \tag{5.79}$$

$$\widetilde{Z}_e = \frac{\widetilde{P}_e}{\widetilde{T}_e}\,\widetilde{u} \tag{5.80}$$

and A, B, C, D, E, F and G are constants given by

$$A = \frac{1}{\tau_f} = \frac{\dot{m}_i}{\rho_0 V_{cc}}, \tag{5.81}$$

$$B = \frac{\Delta H_f}{c_p T_0}, \tag{5.82}$$

$$C = \frac{\widehat{h}}{L_1 c_p \rho_0}, \tag{5.83}$$

$$D = \frac{T_w}{T_0}, \tag{5.84}$$

$$E = \frac{R T_0}{L_{tp} u_0}, \tag{5.85}$$

$$F = \frac{u_0 f}{2 D_{tp}}, \tag{5.86}$$

$$G = \frac{u_0}{L_2}. \tag{5.87}$$

The constants L_1 and L_2 are the two characteristics lengths introduced before. Their definitions are repeated below for convenience:

$$L_1 = \frac{V_{cc}}{A_s}, \tag{5.88}$$

$$L_2 = \frac{V_{cc}}{A_{tp}}. \tag{5.89}$$

Variable	Initial value	Unit
$\widetilde{P}$	1.0	–
$\widetilde{T}$	5.0	–
$\widetilde{u}$	0.0	–
Y_f	0.06	–

Table 5.1: Initial conditions for the numerical evaluation of model equations (5.73)–(5.89). These are the same initial conditions as used by Richards et al. [26].

The adjusted model equations can be implemented numerically without difficulty.

5.3.2 Specifying model parameters and initial conditions

The system of model equations (5.73)–(5.89) is numerically implemented using MATLAB [30] software. The standard MATLAB ODE-solver `ode45`[1] is used to solve the equations. The implementation in MATLAB is straightforward; the codes used can be found in Appendix C.

In order to evaluate the system of model equations (5.73)–(5.89) numerically, the model parameters and suitable initial conditions need to be specified. Where possible, these are chosen equal to those used by Richards et al. The initial conditions and most parameter values are stated clearly in the article of Richards et al., but some critical parameter values are not specified explicitly. This causes some difficulties in trying to reproduce their numerical results. The choices for the model parameters not specified by Richards et al. are discussed below.

The initial conditions used by Richards et al. are given in Table 5.1; they are also used for the numerical evaluation in this report. The values correspond to a combustion chamber that is filled with a stoichiometric fuel-air mixture at ambient pressure, with no outflow and a temperature that is high enough to get the combustion process going immediately. This choice of initial conditions (for the temperature in particular) may be unrealistic, but Richards et al. mention that the choice of initial conditions did not affect the final results.

The model parameters that are specified explicitly by Richards et al., and that are used for the numerical evaluation of the model equations in this report, are given in Table 5.2. Richards et al. based these values on properties of the thermal pulse combustor they used in experiments, running on propane as fuel.

Values for the model parameters that are not specified explicitly by Richards et al., are presented in Table 5.3. The choices for the ambient pressure P_0 and temperature T_0 are natural. The choice for R (equal to that of air) is less natural than it seems; this will be discussed below. Choosing the specific heat, c_p, and the ratio of specific heats, γ, is more difficult, since they depend on the temperature of the mixture in the combustion chamber, as well as on its composition. Note that because of the relation $R = c_p - c_v = c_p \frac{\gamma-1}{\gamma}$ used in the derivation of the model equations, the parameters c_p, γ and R cannot be specified independently of each other. The values chosen for c_p and γ are based

[1]See footnote 2 on page 18

Parameter	Value used	Unit
A'	3.85×10^8	s^{-1}
D_{tp}	0.0178	m
f	0.030	–
$\widehat{h}$	120	$\mathrm{W/m^2/K}$
L_1	0.0119	m
L_2	0.8486	m
L_{tp}	0.6100	m
$\widetilde{T}_{\mathrm{act}}$	50	–
T_w	1000	K
$Y_{f,i}$	0.06	–
τ_f	0.030	s

Table 5.2: Model parameters for the evaluation of model equations (5.73)–(5.89), as specified by Richards et al. in their article [26].

Parameter	Value used	Unit
c_p	1350	$\mathrm{J/kg \cdot K}$
P_0	1.01325×10^5	Pa
R	287	$\mathrm{J/kg \cdot K}$
T_0	300	K
ΔH_f	5×10^7	J/kg
γ	$= c_p/(c_p - R)$	–
ρ_0	$= P_0/RT_0$	$\mathrm{kg/m^3}$

Table 5.3: Model parameters for the numerical evaluation of model equations (5.73)–(5.89) that are not (explicitly) specified by Richards et al. in their article [26].

on those specified in some articles in which the model equations of Richards et al. are used. These choices will be discussed below.

Specifying the gas constant

The gas constant R is chosen to be that of air (i.e., $R = 287\,\mathrm{J/kg\cdot K}$, see Table 2, p.343 of Reference [28]). This seems a natural choice, since it is used in the derivation of the model equations to define the ambient density $\rho_0 = P_0/RT_0$. But it should be noted that it is also used as the gas constant of the mixture in the combustion chamber, as can be seen from the relation $P = \rho RT$. Since the value of the gas constant depends on the composition of the mixture, some kind of average value should be specified. Given that the fuel is propane (C_3H_8), as used by Richards et al., the gas constants for the reactants (R_r) and the products (R_p) are easily calculated from the molecular weights of the compounds forming the mixtures. From the reaction equation

$$C_3H_8 + 5\,(O_2 + 3.76\,N_2) \quad \longrightarrow \quad 3CO_2 + 4\,H_2O\ (g) + 18.8\,N_2, \tag{5.90}$$

it follows that the molecular weights of the reactants (MW_r) and products (MW_p) are given by

$$\begin{aligned} \mathrm{MW}_r &= \chi_{C_3H_8}\mathrm{MW}_{C_3H_8} + \chi_{O_2}\mathrm{MW}_{O_2} + \chi_{N_2}\mathrm{MW}_{N_2} \\ &= \frac{44.094 + 5\cdot 31.999 + 18.8\cdot 28.013}{24.8} = 29.465\,\mathrm{g/mol} \end{aligned} \tag{5.91}$$

and

$$\begin{aligned} \mathrm{MW}_p &= \chi_{CO_2}\mathrm{MW}_{CO_2} + \chi_{H_2O}\mathrm{MW}_{H_2O} + \chi_{N_2}\mathrm{MW}_{N_2} \\ &= \frac{3\cdot 44.010 + 4\cdot 18.015 + 18.8\cdot 28.013}{25.8} = 28.323\,\mathrm{g/mol}. \end{aligned} \tag{5.92}$$

Here χ_X denotes the molal fraction of a compound X. The values for the molecular weights of the compounds are obtained from Reference [28, Table 2, p.344]. The gas constants are given by $R_r = R_u/\mathrm{MW}_r = 282.2\,\mathrm{J/kg\cdot K}$ and $R_p = R_u/\mathrm{MW}_p = 293.6\,\mathrm{J/kg\cdot K}$, where $R_u = 8.31441\,\mathrm{J/mol\cdot K}$ is the universal gas constant (see Reference [28, in back]). The gas constant of a reactants-products mixture in the reaction chamber will lie somewhere in between these two values. It is approximately equal to the gas constant of the ambient environment. Thus, the chosen value for R is rather convenient: a different choice would yield a scaling density ρ_0 different from that of the ambient environment.

Specifying the specific heat and ratio of specific heats

Choosing proper values for the specific heat (c_p) and the associated ratio of specific heats ($\gamma = \frac{c_p}{c_p - R}$) is more complicated, since they depend on the temperature as well as on the composition of the mixture in the combustion chamber. The variation with the temperature of the specific heats of the products and reactants mixtures can be obtained from literature. Table 3s, p.346, from Reference [28] shows the temperature dependency of the molal specific heats at constant pressure, $\bar{c}_p$, for various gases. From this table, the molal specific heat

$$\bar{c}_p = a + bT + cT^2 + dT^3, \quad [T \text{ in K}, \bar{c}_p \text{ in J/mol} \cdot \text{K}]$$

Mixture	a	b	c	d
Reactants	26.88	1.416×10^{-2}	-0.1655×10^{-5}	-0.6336×10^{-9}
Products	28.65	0.6108×10^{-2}	0.3453×10^{-5}	-1.782×10^{-9}

Table 5.4: Temperature dependency of molal specific heats at constant pressure for the reactants and products. Range of validity: 273–1500 K. The values are compiled from Table 3s, p.346, in Reference [28].

of a mixture of compounds, $\bar{c}_{p,\text{mix}}$ can be obtained by using the relation (see Reference [28] for instance):

$$\bar{c}_{p,\text{mix}} = \sum_i \chi_i \bar{c}_{p,i}. \tag{5.93}$$

Here the sum is taken over all compounds of the mixture and χ_i and $\bar{c}_{p,i}$ are the corresponding mass fractions and molal specific heats, respectively. The resulting (polynomial) expressions in temperature for the molal specific heats of the reactants and products are given in Table 5.4. The specific heat at constant pressure of a mixture ($c_{p,\text{mix}}$) can be computed from the molal specific heat ($\bar{c}_{p,\text{mix}}$) by using the relation

$$c_{p,\text{mix}} = \frac{\bar{c}_{p,\text{mix}}}{\text{MW}_{\text{mix}}}, \tag{5.94}$$

where MW_{mix} is the mixture's molecular weight.

Figure 5.2 shows plots of the specific heats at constant pressure for the reactants and products over the temperature range 273–1500 K, the range of validity of the values from the consulted reference. The plots show that the specific heats vary significantly with the temperature. In the specified temperature range, the specific heat rises roughly by 30 J/kg · K for every 100 K the temperature rises.

The mean temperature in the combustion chamber is not known beforehand. It will lie somewhere in between 300 K, the temperature of the reactants upon entrance, and 2267 K, the adiabatic flame temperature of a stoichiometric propane-air mixture (see Table B.1, p.543, in Reference [33]). Without additional information on the combustion process and associated temperatures it is difficult to make a sensible choice for the model parameter c_p.

Choosing parameters based on an average temperature from the numerical results of Richards et al. did not yield the same numerical results. Also, the numerical solution of the model equations is rather sensitive to the choice of the model parameters. Since several model parameters need to be 'guessed' in order to reproduce the numerical results of Richards et al., this means that a process of trial and error is laborious. In the end, the model parameters used for the numerical evaluation of the model equations were obtained from articles [8] (co-authored by Richards) and [12], in which the model of Richards et al. is used.

Both the articles use $c_p = 1200$ J/kg · K and $\gamma = 1.27$. Using the relation $R = c_p \frac{\gamma - 1}{\gamma}$ gives $R = 255.1$ J/kg · K for the gas constant, nowhere near the value for the mixture in the combustion chamber computed above. Besides, the numerical results when evaluating the model equations with these parameters

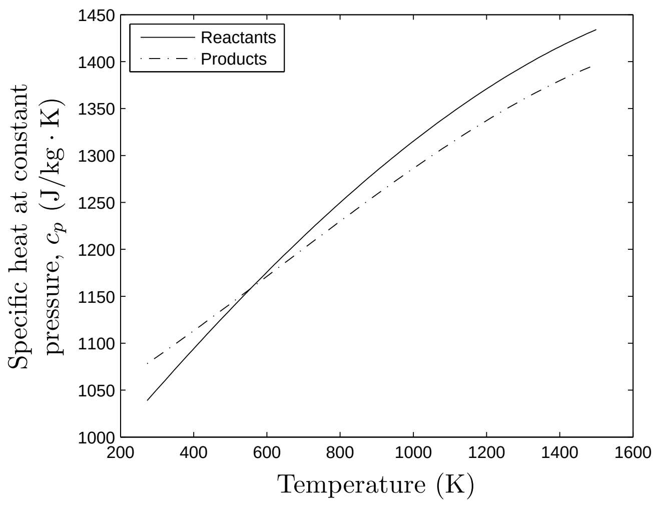


Figure 5.2: Temperature dependency of specific heats at constant pressure of the reactants and products.

are different from those of Richards et al. The specific heat associated with $\gamma = 1.27$ and $R = 287\,\text{J/kg} \cdot \text{K}$ is given by $c_p = \frac{R\gamma}{\gamma - 1} = 1350.0\,\text{J/kg} \cdot \text{K}$. Using these values for the numerical evaluation of the model equations yields results very similar to those of Richards et al., if $\Delta H_f = 5 \times 10^7$ J/kg is used for the heat of combustion. Table B.1, p.543, in Reference [33] gives $\Delta H_f = 4.6357 \times 10^7$ J/kg for the heat of combustion (lower heating value) of propane, so the value used for obtaining the numerical results could be a rough approximation. It should be noted that the numerical solution of the model equations showed a very different behavior for values of ΔH_f less than 4.88×10^7 J/kg, showing the sensitivity to the model parameters. The article [19] (also co-authored by Richards), found after the numerical results presented in this section were already obtained, also uses these model parameter values (for γ, c_p, R and ΔH_f). This makes it more likely that the values mentioned above were indeed used by Richards et al. in their original article.

A remark on the parameter choices is in order. In none of the articles is the choice for the model parameters c_p and γ motivated. In fact, the choices do not seem to correspond to the mean temperature in the combustion chamber that the numerical evaluation of the model equations yields. Also, the numerical solution of the model equations is sensitive to these parameters. This raises the question whether the solution to the model equations would not show a very different behavior if the temperature dependency of c_p and γ were accounted for.

5.4 Numerical results and discussion

The model equations were solved with the MATLAB-codes presented in Appendix C. This appendix also contains all the codes that were used to produce the plots

Chapter 6

Conclusions and research questions

The main goal of the research project is to get a better understanding of how the physical processes involved in (aerovalved) pulse combustion affect the operation of a pulse combustor through numerical modeling. To this end, we have evaluated three lumped parameter models. In our opinion, for a sufficiently realistic description of an (aerovalved) pulse combustor, these models lack two important features. Firstly, a more detailed description of the wave dynamics is needed. Secondly, they do not model the effect of an *aerovalve* on the operation of a pulse combustor. While aerovalved pulse combustion has been modeled in the work of Narayanaswami and Richards [19] and of Richards and Gemmen [25] by extending the model of Richards et al. (which was discussed in Chapter 5), it still results in a lumped parameter model, that provides little insight into the wave dynamics. Therefore, the development of a one-dimensional model, capable of providing an approximation to the wave dynamics inside an aerovalved pulse combustor, is the first step in the follow-up research.

The one-dimensional modeling of aerovalved pulse combustion can be divided into a number of submodeling problems.

- The one-dimensional equations of gas dynamics need to be formulated, including energy sources (from combustion) and energy sinks and dissipative processes (such as heat transfer, wall friction). These describe the wave dynamics in the system, driven by combustion and damped by dissipative processes.
- The heat released by combustion needs to be quantified. Factors of influence on this are the mixing processes of fuel and oxidizer, the mixing of reactants and hot products, and a chemical reaction rate. In turn, the mixing processes depend on the flow field and injection characteristics.
- The geometry of the pulse combustor needs to be specified. It will be considered as a continuous volume, including aerovalve, combustion chamber and tailpipe, in which the wave dynamics are described.
- Finally, boundary conditions, describing the effects of the aerovalve inlet and the tailpipe exit, need to be specified.

These submodels will be taken from literature, with the model presented by Barr and Dwyer [4] as a starting point. In the modeling process, the temperature dependency of the specific heat needs to be considered. To gain insight into the pulse combustion process, the contributions of the various physical terms to the energy of the oscillations need to be quantified.

The main focus of the follow-up research will be on developing a numerical scheme, that is capable of approximating the solution of the model equations for given geometry and boundary conditions with sufficient accuracy. Of course, a simple mathematical model will not be able to predict the exact behavior of a real pulse combustor, so it may be questioned how accurate the numerical method needs to be. However, the purpose of the research project is to gain insight into how the various physical processes affect the operation of a pulse combustor. Therefore, it is desirable that the calculated contributions of the various physical processes to the operation characteristics are not affected by artificial effects introduced by the numerical model. Specifically:

- **Numerical dispersion.** Phase differences between heat release by combustion and the acoustic waves drive the pulse combustion process, and it is very sensitive to these differences. Therefore, the numerical method needs to limit the numerical dispersion.

- **Numerical dissipation.** Damping processes affect the amplification of the system's acoustics, which may significantly affect the amplitude of the oscillations in the system. Also, damping processes may affect the system's frequency, which in turn affects the phase difference between heat release and the acoustic waves. Therefore, the numerical method also needs to limit the numerical dissipation.

- **Numerical accuracy.** Finally, the nature of pulse combustion operation is in the amplification and maintenance of disturbances, by favorably linking heat release to resonant acoustics. Thus, a numerical method for solving model equations that capture this nature, will be susceptible to (numerical) disturbances. Therefore, the numerical method needs to be sufficiently accurate.

Of course, the important question is: How accurate is 'sufficiently accurate,' and how much numerical dispersion and dissipation may be allowed? Also, are these properties of a numerical scheme indeed as important as suggested above? These questions are important elements in the verification of the numerical model (i.e., the process of establishing whether the numerical model accurately represents the model equations). For verification purposes, benchmark problems and/or problems with an analytical solution need to be selected.

The final question is: Is the model capable of capturing the behavior of a pulse combustor, at least in a qualitative sense? Of course, a one-dimensional model cannot describe the pulse combustion process accurately. Conditions for which the one-dimensional approximation may be acceptable for describing (the characteristics of) the pulse combustion process, need to be established. The geometry of the modeled pulse combustor can be expected to be a major factor in this respect. To address the final question, the model should be validated against experimental results (of course under the conditions for which the model is intended).

Bibliography

[1] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions: with formulas, graphs and mathematical tables.* Dover Publications, Inc., New York, 1965. For an online copy of the tenth printing (1972) of this work, see web-site `http://www.math.sfu.ca/~cbm/aands/`.

[2] Frederick W. Ahrens, Choong Kim, and Shiu-Wing Tam. An analysis of the pulse combustion burner. *ASHRAE Transactions*, 84, Part 1:488–507, 1978.

[3] P. K. Barr, J. O. Keller, T. T. Bramlette, C. K. Westbrook, and J. E. Dec. Pulse combustor modeling demonstration of the importance of characteristic times. *Combustion and Flame*, 82:252–269, 1990.

[4] Pamela K. Barr and Harry A. Dwyer. *Pulse Combustor Dynamics: A Numerical Study*, chapter 22, pages 673–710. Volume 135 of Oran and Boris [21], 1991.

[5] Paul F. Byrd and Morris D. Friedman. *Handbook of Elliptic Integrals for Engineers and Physicists.* Die Grundlehren der Mathematischen Wissenschaften, Band LXVII. Springer-Verlag, Berlin, 1954.

[6] Boa-Teh Chu. On the energy transfer to small disturbances in fluid flow (Part I). *Acta Mechanica*, 1(3):215–234, September 1965. (Printed.).

[7] Subhashis Datta, Achintya Mukhopadhyay, and Dipankar Sanyal. Modeling and analysis of the nonlinear dynamics of a thermal pulse combustor. *42nd AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, 2006. 9–12 July 2006, Sacramento, California. American Institute of Aeronautics and Astronautics, Paper AIAA 2006-4396.

[8] C. Stuart Daw, John F. Thomas, George A. Richards, and Lakshmanan L. Narayanaswami. Chaos in thermal pulse combustion. *Chaos*, 5(4):662–670, 1995.

[9] John E. Dec and Jay O. Keller. Pulse combustor tail-pipe heat-transfer dependence on frequency, amplitude, and mean flow rate. *Combustion and Flame*, 77:359–374, 1989.

[10] John E. Dec, Jay O. Keller, and Vedat S. Arpaci. Heat transfer enhancement in the oscillating turbulent flow of a pulse combustor tail pipe. *International Journal of Heat and Mass Transfer*, 35(9):2311–2325, 1992.

[11] H. M. Heravi, J. R. Dawson, P. J. Bowen, and N. Syred. Primary pollutant prediction from integrated thermofluid-kinetic pulse combustor models. *Journal of Propulsion and Power*, 21(6):1092–1097, November-December 2005.

[12] Visarath In, Mark L. Spano, Joseph D. Neff, William L. Ditto, C. Stuart Daw, K. Dean Edwards, and Ke Nguyen. Maintenance of chaos in a computational model of a thermal pulse combustor. *Chaos*, 7(4):605–613, 1997.

[13] Dominic William Jordan and Peter Smith. *Nonlinear ordinary differential equations.* Oxford applied mathematics and computing series. Oxford University Press Inc., New York, 1994.

[14] J. O. Keller, T. T. Bramlette, J. E. Dec, and C. K. Westbrook. Pulse combustion: The importance of characteristic times. *Combustion and Flame*, 75:33–44, 1989.

[15] J. O. Keller, T. T. Bramlette, C. K. Westbrook, and J. E. Dec. Pulse combustion: The quantification of characteristic times. *Combustion and Flame*, 79:151–161, 1990.

[16] J. O. Keller and I. Hongo. Pulse combustion: The mechanisms of NO_x production. *Combustion and Flame*, 80:219–237, 1990.

[17] Ali Kilicarslan. Frequency evaluation of a gas-fired pulse combustor. *International Journal of Energy Research*, 29:439–454, 2005.

[18] D. Kretschmer and J. Odgers. Modeling of gas turbine combustors—a convenient reaction rate equation. *Journal of Engineering for Power (Transactions of the ASME)*, pages 173–180, July 1972.

[19] L. Narayanaswami and G. A. Richards. Pressure-gain combustion: Part I—Model development. *Journal of Engineering for Gas Turbines and Power*, 118(3):461–468, July 1996.

[20] F. Nicoud and T. Poinsot. Thermoacoustic instabilities: Should the Rayleigh criterion be extended to include entropy changes? *Combustion and Flame*, 142:153–159, 2005.

[21] Elaine S. Oran and Jay P. Boris, editors. *Numerical Approaches to Combustion Modeling*, volume 135 of *Progress in Astronautics and Aeronautics.* The American Institute of Aeronautics and Astronautics, Inc., Washington, DC, 1991.

[22] Sungbae Park, Anuradha Annaswamy, and Ahmed Ghoniem. Heat release dynamics modeling of kinetically controlled burning. *Combustion and Flame*, 128:217–231, 2002.

[23] A. A. Putnam, F. E. Belles, and J. A. C. Kentfield. Pulse combustion. *Progress in Energy and Combustion Science*, 12:43–79, 1986.

[24] Rayleigh. The explanation of certain acoustical phenomena. *Nature*, 18(455):319–321, July 1878.

[25] G. A. Richards and R. S. Gemmen. Pressure-gain combustion: Part II—Experimental and model results. *Journal of Engineering for Gas Turbines and Power*, 118(3):469–473, July 1996.

[26] G. A. Richards, G. J. Morris, D. W. Shaw, S. A. Keeley, and M. J. Welter. Thermal pulse combustion. *Combustion Science and Technology*, 94:57–85, 1993.

[27] H. Schlichting. *Boundary Layer Theory*. McGraw-Hill, New York, 1979.

[28] Philip S. Schmidt, Ofodike A. Ezekoye, John R. Howell, and Derek K. Baker. *Thermodynamics: An integrated learning system*. John Wiley & Sons, Inc., New York, 2006.

[29] Lawrence F. Shampine and Mark W. Reichelt. The MATLAB ODE Suite. *SIAM Journal of Scientific Computing*, 18(1):1–22, January 1997.

[30] The MathWorks, Inc. MATLAB, version 7.3.0.298 (R2006b), 2006. Web-site: `http://www.mathworks.com`.

[31] Philip A. Thompson. *Compressible-Fluid Dynamics*. Advanced Engineering Series. McGraw-Hill, Inc., New York, 1972.

[32] N. A. W. Tiemessen and R. J. M. Bastiaans. Aerovalved pulse combustion. Bachelor's Thesis Project, Eindhoven University of Technology, Eindhoven, October 2006. (In Dutch.).

[33] Stephen R. Turns. *An Introduction to Combustion: Concepts and applications*. McGraw-Hill Series in Mechanical Engineering. McGraw-Hill, Inc., New York, 1996.

[34] D. J. van Mullem and R. J. M. Bastiaans. Modelleren van een pulse combustor. Bachelor's Thesis Project, Eindhoven University of Technology, Eindhoven, January 2007. (In Dutch.).

Appendix A

MATLAB codes for model of Kilicarslan

In this chapter, the system of ordinary differential equations (ODEs) of Kilicarslan's model (see (3.16)–(3.22)) is numerically implemented using MATLAB [30] software. The standard MATLAB ODE solver `ode45` is used to solve the equations. It is assumed that the user is familiar with MATLAB, and the ODE solver.

The MATLAB ODE solvers for initial value problems (a.o. the solver `ode45`) can be used to solve systems of ODEs written in the general form

$$\frac{d\mathbf{y}}{dt} = \mathbf{f}(t, \mathbf{y}), \tag{A.1}$$

$$\mathbf{y}(0) = \mathbf{y}_0, \tag{A.2}$$

where $\mathbf{y}$ is a vector, $\mathbf{f}$ is a vector function, t is time, and $\mathbf{y}_0$ is a vector of initial values. To obtain a numerical solution, a function implementing the right-hand side of (A.1) must be passed to the solver, together with the initial values vector $\mathbf{y}_0$. The function that is passed to the solver, should return a column vector.

In Section A.1 the model equations of Kilicarslan are written in the general form of (A.1) and (A.2), and the function that can be passed to the solver is given. In Section A.2 the code is given that was used to generate the data for the figures of Section 3.5. Finally, Section A.3 presents the code that was used to plot the figures of Section 3.5, using the generated data.

It should be mentioned that the code was written rather quickly, and no attempt has been made to optimize it. In the implementation of the functions, no attention is paid to checking whether the input arguments are of the correct form. Thus, the user should take care when prescribing the input arguments, although eventually an error will be produced if the input is of an unexpected form. Also, plots produced by the code of Section A.3 may differ somewhat from the figures in Section 3.5. This is because they were manipulated manually (e.g. by resizing them, replacing labels) before including them in the report.

A.1 Numerical implementation

The model equations of Kilicarslan (see (3.16)–(3.22)) can be written in the general form of (A.1) and (A.2) by defining $\mathbf{y}$ and $\mathbf{f}$ in the following way:

$$\mathbf{y} = \begin{pmatrix} p \\ \dot{m}_e \end{pmatrix}, \tag{A.3}$$

$$\mathbf{f}(t, \mathbf{y}) = \begin{pmatrix} A F(y_1) - B\, y_2 \\ C\, y_1 \end{pmatrix}, \tag{A.4}$$

where A, B and C, expressed in the model parameters, are:

$$\begin{aligned} A &= k_g(1+r)\frac{(\gamma-1)h_e}{V_{cc}}, && \text{(A.5)} \\ B &= \frac{(\gamma-1)h_e}{V_{cc}}, && \text{(A.6)} \\ C &= \frac{A_{tp}}{L_{tp}}, && \text{(A.7)} \\ F(p) &= \sqrt{|p|}\,\mathbf{1}_{(-\infty,0)}(p). && \text{(A.8)} \end{aligned}$$

The function that must be passed to the solver (implementing $\mathbf{f}(t, \mathbf{y})$), depends on the model parameters. To be able to vary all of the model parameters, it was chosen to pass a structure of model parameters to that function. The structure has fields in which the model parameter values are stored; the fieldnames correspond to the model parameters. Such a structure can be generated with help of the function `setKilicarslan` (see Listing A.1.1 at the end of this section). Calling the function without any input arguments returns a structure with the model parameters set to their default values. Calling the function with input of the form `'par1',val1,...` (i.e. in pairs of fieldnames and values), sets the model parameters specified by the fieldnames `par`n to the corresponding values `val`n, while the other parameters are set to their default values. Together, Tables A.1 and A.2 list the model parameters that are stored in a parameter structure, their fieldnames, and the default values.

Not all model parameters can be specified independently: those that can, are listed in Table A.1, while those that depend on others are listed in Table A.2. While no error is produced if the input of the function `setKilicarslan` contains one of the dependent model parameters, it will have no effect on the output. Its value in the resulting structure will just be the value specified by the definition given in Table A.2, which can be rewritten using solely independent parameters. Of course, it is possible to generate, or manipulate, a model parameter structure directly, but in doing so, one risks that the dependent model parameter values are inconsistent with the independent parameter values. Thus, it is advised to set the model parameters using the function `setKilicarslan`.

The function `setKilicarslan` uses the auxiliary function `setparam`, which sets specified fields of a given structure to specified values. In the process, it checks whether the specified fields exist in the structure, and it gives an error if that is not the case. Since the function `setparam` is also used to set parameters in the numerical implementation of the models of Ahrens et al. (see Appendix B) and Richards et al. (see Appendix C), its implementation and usage is discussed in Appendix D.

Parameter	Fieldname	Default value	Unit
$A_{v,g}$	`Avg`	6.26×10^{-5}	m^2
$C_{D,g}$	`CDg`	0.6	–
c_p	`cp`	1009.7	J/kg/K
ΔH_f	`DHf`	50.016×10^6	J/kg
D_{tp}	`Dtp`	0.042	m
L_{tp}	`Ltp`	2	m
MW_g	`MWg`	16.043	kg/kmol
P_0	`P0`	1.01325×10^5	Pa
r	`r`	17.12	–
R_a	`Ra`	287	J/kg/K
R_u	`Ru`	8.31441×10^3	J/kmol/K
T_0	`T0`	300	K
V_{cc}	`Vcc`	2.609705×10^{-3}	m^3

Table A.1: Independent model parameters of Kilicarslan's model, the fieldnames under which they are stored in the parameter structure, their default values, and the units in which they are expressed.

Parameter	Fieldname	Defined by	Unit
A_{tp}	`Atp`	$= \pi D_{tp}^2/4$	m^2
γ	`gam`	$= \frac{c_p}{c_p - R_a}$	–
h_e	`he`	$= h_r + \frac{\Delta H_f}{1+r}$	J/kg
h_r	`hr`	$= c_p T_0$	J/kg
k_g	`kg`	$= \sqrt{2\rho_g} C_{D,g} A_{v,g}$	$\mathrm{kg}^{1/2}\,\mathrm{m}^{1/2}$
R_g	`Rg`	$= R_u/\mathrm{MW}_g$	J/kg/K
ρ_g	`rhog`	$= \frac{P_0}{R_g T_0}$	$\mathrm{kg/m}^3$

Table A.2: Dependent model parameters of Kilicarslan's model, the fieldnames under which they are stored in the parameter structure, expressions defining their values, and the units in which they are expressed.

Function	Input	Description
`odeKilicarslan`	`t,y,s`	Returns the value of the function $\mathbf{f}(t, \mathbf{y})$ of (A.4), for specified time `t`, vector `y`, and the model parameters stored in structure `s`.
`setKilicarslan`	none	Returns a parameter structure with *default* values, that can be passed to the function `odeKilicarslan`.
	`'par1',val1,...`	Returns a parameter structure with the *independent* parameters `par`n set to the specified values `val`n; the other parameters have default values. The structure can be passed to the function `odeKilicarslan`.
`setparam`	`s,'fld1',val1,...`	See Appendix D.

Table A.3: Overview of the functions that are defined in Section A.1.

The function `odeKilicarslan` is an implementation of the function $\mathbf{f}(t, \mathbf{y})$ of (A.4). Besides the time t and the vector $\mathbf{y}$, the implemented function also needs a parameter structure specifying the model parameters as input. The code of the function `odeKilicarslan` is given in Listing A.1.2 at the end of this section.

An overview of the MATLAB functions defined in this section, is presented in Table A.3.

Listing A.1.1: Function setKilicarslan. Given a paired list of fieldnames and values, it returns a structure of model parameters, which can be passed to the function `odeKilicarslan`. Model parameter values that are not specified, are set to their default values.

setKilicarslan.m

```
function sOut = setKilicarslan(varargin);

% Set default values

s = struct(...
    'Atp' , []          ,... % (calculated, see below)
    'Avg' , 6.26e-5     ,... % [ m^2 ]
    'CDg' , 0.6         ,... % [ - ]
    'cp'  , 1009.7      ,... % [ J/kg/K ]
    'DHf' , 50.016e+6   ,... % [ J/kg ]
    'Dtp' , 0.042       ,... % [ m ]
    'gam' , []          ,... % (calculated, see below)
    'he'  , []          ,... % (calculated, see below)
    'hr'  , []          ,... % (calculated, see below)
    'kg'  , []          ,... % (calculated, see below)
    'Ltp' , 2           ,... % [ m ]
    'MWg' , 16.043      ,... % [ kg/kmol ]
    'P0'  , 1.01325e+5  ,... % [ Pa ]
    'r'   , 17.12       ,... % [ - ]
    'Ra'  , 287         ,... % [ J/kg/K ]
    'Rg'  , []          ,... % (calculated, see below)
    'rhog', []          ,... % (calculated, see below)
    'Ru'  , 8.31441e+3  ,... % [ J/kmol/K ]
    'T0'  , 300         ,... % [ K ]
    'Vcc' , 2.609705e-3 );   % [ m^3 ]

% Set user specified model parameters

if nargin > 0
    s = setparam(s,varargin{:});
end

% Calculate dependent model parameters

s.Atp = 0.25*pi*s.Dtp^2;         % [ m^2 ]
s.gam = s.cp/(s.cp - s.Ra);      % [ - ]
s.hr  = s.cp*s.T0;               % [ J/kg ]
s.he  = s.hr + s.DHf/(1 + s.r);  % [ J/kg ]
s.Rg  = s.Ru/s.MWg;              % [ J/kg/K ]

s.rhog = s.P0/(s.Rg*s.T0);            % [ kg/m^3 ]
s.kg   = sqrt(2*s.rhog)*s.CDg*s.Avg;  % [ kg^0.5 m^0.5 ]

% Assign output structure

sOut = s;
```

setKilicarslan.m

Listing A.1.2: Function odeKilicarslan. Given the following input arguments: time t, the vector $\mathbf{y}$ (see (A.3)), and a model parameter structure, it returns the vector $\mathbf{f}(t, \mathbf{y})$ defined by (A.4).

odeKilicarslan.m

```
function dydt = odeKilicarslan(t,y,s);

% Compute auxiliary constants

B = (s.gam - 1)/s.Vcc*s.he;
A = s.kg*(1 + s.r)*B;
C = s.Atp/s.Ltp;

F = @(p) sqrt(abs(min(p,0)));  % function of gauge pressure, p

% Compute dydt
%    y(1) == gauge pressure, [ Pa ]
%    y(2) == mass flux at chamber exit, [ kg/s ]

dy1dt = A*F(y(1)) - B*y(2);
```

```
16  dy2dt = C*y(1);
17
18  dydt  = [dy1dt; dy2dt];
```

odeKilicarslan.m

A.2 Generating data for graphs

The script file `dataKilicarslan.m` (see Listing A.2.1) contains the code that was used to generate the data for the figures of Section 3.5. By setting the Matlab variable `identifier` in Line 16 to a certain string and running the script, data for the figure associated with the identifier are generated, and stored under an appropriate name. The available choices for the identifier are listed in Table A.4, together with descriptions of the data that are generated for those choices and the filenames under which the data are stored. The data that are stored in the `.mat` files are: the model parameter structure and the initial values vector that were passed to the ODE solver `ode45`, and the solution structure that was returned by the solver. The initial conditions that were used, are given in Table A.5.

Listing A.2.1: Script file dataKilicarslan.m. Generates data for the figures of Section 3.5. The choice of the identifier set in Line 16 determines the data that are generated.

dataKilicarslan.m

```
1   % dataKilicarslan.m
2
3   % Clear workspace, close figure windows, and clear command window
4
5   clear all
6   close all
7   clc
8
9   % Set identifier for generation of data for a certain figure from
10  % the Interim Report, Chapter 3. Choose from:
11  %
12  %   'Figure3_3'  = default parameter values; no oscillations
13  %   'Figure3_4'  = parameter values adjusted to show some
14  %                  oscillations
15
16  identifier = 'Figure3_3';
17
18  % Set initial values for ODE-solver
19  %    y(1) == gauge pressure, [ Pa ]
20  %    y(2) == mass flux at chamber exit, [ kg/s ]
21
22  y1ini = 1.01325e+5;  % [ Pa ]
23  y2ini = 0;           % [ kg/s ]
24
25  y0 = [y1ini; y2ini];
26
27  % Set options for ODE-solver
```

Identifier	Filename	Description
`'Figure3_3'`	`dataFigure3_3.mat`	Generates data for Figure 3.3. Default parameter values are used.
`'Figure3_4'`	`dataFigure3_4.mat`	Generates data for Figure 3.4. Three parameter values are changed from their defaults to get an oscillatory solution.

Table A.4: Identifier choices for the script file `dataKilicarslan.m` (see Line 16 of Listing A.2.1), and the filenames under which the generated data are stored. The following data are stored in the `.mat` files: the parameter structure (`s`), the initial values vector (`y0`), and the solution structure (`sol`) that is returned by the ODE solver.

Initial condition		Value used	Unit
`y0(1)`	$= p(0)$	1.01325×10^5	Pa
`y0(2)`	$= \dot{m}_e(0)$	0	kg/s

Table A.5: Initial values, stored in the vector `y0` that was passed to the ODE solver to generate the data for the figures of Section 3.5.

```

options = odeset('AbsTol',1e-8,'RelTol',1e-6);

% Set time-span for ODE-solver

tmin  = 0;     % [ s ]
tmax  = 0.15;  % [ s ]
tspan = [tmin tmax];

% Initialize structure of model parameters (default values)

s = setKilicarslan;

% Select data to generate for plotting the figure, and save it

switch identifier

    case 'Figure3_3'

        % Run ODE-solver with default parameter values, and save

        sol = ode45(@odeKilicarslan,tspan,y0,options,s);
        save(['data',identifier],'s','y0','sol')

    case 'Figure3_4'

        % Run ODE-solver with parameter values adjusted to show
        % some oscillations, and save

        s = setKilicarslan(...
            'Avg', 0.5  * s.Avg ,...
            'Ltp', 0.25 * s.Ltp ,...
            'Vcc', 10   * s.Vcc );
        sol = ode45(@odeKilicarslan,tspan,y0,options,s);
        save(['data',identifier],'s','y0','sol')

    otherwise

        error('Unknown identifier choice.');

end  % switch
```
dataKilicarslan.m

A.3 Plotting graphs

The script file `plotKilicarslan.m` (see Listing A.3.1) contains the code that was used to plot the figures of Section 3.5, using the data generated with the script `dataKilicarslan` (see Section A.2). By setting the MATLAB variable `identifier` in Line 15 to a certain string and running the script, data for the figure associated with that identifier is loaded, and a graph is plotted and saved under an appropriate name. The available choices for the identifier are listed in Table A.6. It mentions which figures are produced for those choices, and

Identifier	Filename ⟨needed⟩	Description
`'Figure3_3'`	`Figure3_3.fig` ⟨`dataFigure3_3.mat`⟩	Plots Figure 3.3.
`'Figure3_4'`	`Figure3_4.fig` ⟨`dataFigure3_4.mat`⟩	Plots Figure 3.4.

Table A.6: Identifier choices for the script file `plotKilicarslan.m` (see Line 15 of Listing A.3.1). The figures are saved as `.fig` files. Data that is needed to plot the figures, is loaded from the `.mat` files; they are indicated by angular brackets.

the filenames under which the figures are saved (i.e. the `.fig` files). The data files (extension: `.mat`) that are needed to plot the figures, are shown between angular brackets.

Listing A.3.1: Script file plotKilicarslan.m. Plots the figures from Section 3.5, using data generated with `dataKilicarslan.m` (see Section A.2). The choice of the identifier set in Line 15 determines the graph that is plotted.

plotKilicarslan.m

```
1   % plotKilicarslan.m
2
3   % Clear workspace, close figure windows, and clear command window
4
5   clear all
6   close all
7   clc
8
9   % Set identifier to plot a certain graph from the Interim Report,
10  % Chapter 3. Choose from:
11  %
12  %   'Figure3_3'  = plot Figure 3.3 from Interim Report
13  %   'Figure3_4'  = plot Figure 3.4 from Interim Report
14
15  identifier = 'Figure3_3';
16
17  % Select data to load, plot requested figure, and save it
18
19  switch identifier
20
21      case 'Figure3_3'
22
23          % Load data
24
25          load 'dataFigure3_3';
26
27          % Extract data for time and gauge pressure from solution
28          % structure
29
30          t = sol.x.';
31          y = sol.y.';
```

```
        p = y(:,1);      % gauge pressure, [ Pa ]

        % Plot graph, annotate, and save

        h = figure('Name','Figure 3.3 of Interim Report');
        plot(t,p*1e-3)  % time in [ s ], gauge pressure in [ kPa ]

        title('Figure 3.3')
        xlabel('Time (s)')
        ylabel('Gauge pressure (kPa)')
        xlim([0 max(t)])
        grid on

        saveas(h,'Figure3_3','fig')

    case 'Figure3_4'

        % Load data

        load 'dataFigure3_4';

        % Extract data for time and gauge pressure from solution
        % structure

        t = sol.x.';
        y = sol.y.';
        p = y(:,1);      % gauge pressure, [ Pa ]

        % Plot graph, annotate, and save

        h = figure('Name','Figure 3.4 of Interim Report');
        plot(t,p*1e-3)  % time in [ s ], gauge pressure in [ kPa ]

        title('Figure 3.4')
        xlabel('Time (s)')
        ylabel('Gauge pressure (kPa)')
        xlim([0 max(t)])
        grid on

        saveas(h,'Figure3_4','fig')

    otherwise

        error('Unknown identifier choice.');

end  % switch
```

plotKilicarslan.m

Appendix B

MATLAB codes for model of Ahrens et al.

In this chapter, the system of ordinary differential equations (ODEs) of the model of Ahrens et al. (see (4.12)–(4.19)) is numerically implemented using MATLAB [30] software. The standard MATLAB ODE solver `ode45` is used to solve the equations. It is assumed that the user is familiar with MATLAB, and the ODE solver.

The MATLAB ODE solvers for initial value problems (a.o. the solver `ode45`) can be used to solve systems of ODEs written in the general form

$$\frac{d\mathbf{y}}{dt} = \mathbf{f}(t, \mathbf{y}), \tag{B.1}$$

$$\mathbf{y}(0) = \mathbf{y}_0, \tag{B.2}$$

where $\mathbf{y}$ is a vector, $\mathbf{f}$ is a vector function, t is time, and $\mathbf{y}_0$ is a vector of initial values. To obtain a numerical solution, a function implementing the right-hand side of (A.1) must be passed to the solver, together with the initial values vector $\mathbf{y}_0$. The function that is passed to the solver, should return a column vector.

In Section B.1 the model equations of Ahrens et al. are written in the general form of (B.1) and (B.2), and the function that can be passed to the solver is given. In Section B.2 the code is given that was used to generate the data for the figures of Section 4.6. Finally, Section B.3 presents the code that was used to plot the figures of Section 4.6, using the generated data.

It should be mentioned that the code was written rather quickly, and no attempt has been made to optimize it. In the implementation of the functions, no attention is paid to checking whether the input arguments are of the correct form. Thus, the user should take care when prescribing the input arguments, although eventually an error will be produced if the input is of an unexpected form. Also, plots produced by the code of Section B.3 may differ somewhat from the figures in Section 4.6. This is because they were manipulated manually (e.g. by resizing them, replacing labels) before including them in the report.

B.1 Numerical implementation

The model equations of Ahrens et al. (see (4.12)–(4.19)) can be written in the general form of (B.1) and (B.2) by defining $\mathbf{y}$ and $\mathbf{f}$ in the following way:

$$\mathbf{y} = \begin{pmatrix} p \\ \dot{m}_e \end{pmatrix}, \tag{B.3}$$

$$\mathbf{f}(t, \mathbf{y}) = \begin{pmatrix} AF(y_1) + B(y_1 + P_0) - C\, y_2 \\ D\, y_1 \end{pmatrix}, \tag{B.4}$$

where A, B, C and D, expressed in the model parameters, are:

$$\begin{aligned} A &= k_g(1+r)\frac{\gamma-1}{V_{cc}}h_r & \text{(B.5)} \\ B &= \frac{A_b U_f}{RT_0}\frac{\gamma-1}{V_{cc}}\frac{\Delta H_f}{1+r}, & \text{(B.6)} \\ C &= \frac{(\gamma-1)h_e}{V_{cc}}, & \text{(B.7)} \\ D &= \frac{A_{tp}}{L_{tp}}, & \text{(B.8)} \\ F(p) &= \sqrt{|p|}\,\mathbf{1}_{(-\infty,0)}(p). & \text{(B.9)} \end{aligned}$$

The function that must be passed to the solver (implementing $\mathbf{f}(t, \mathbf{y})$), depends on the model parameters. To be able to vary all of the model parameters, it was chosen to pass a structure of model parameters to that function. The structure has fields in which the model parameter values are stored; the fieldnames correspond to the model parameters. Such a structure can be generated with help of the function `setAhrensEa` (see Listing B.1.1 at the end of this section). Calling the function without any input arguments returns a structure with the model parameters set to their default values. Calling the function with input of the form `'par1',val1,...` (i.e. in pairs of fieldnames and values), sets the model parameters specified by the fieldnames `par`*n* to the corresponding values `val`*n*, while the other parameters are set to their default values. Together, Tables B.1 and B.2 list the model parameters that are stored in a parameter structure, their fieldnames, and the default values.

Not all model parameters can be specified independently: those that can, are listed in Table B.1, while those that depend on others are listed in Table B.2. While no error is produced if the input of the function `setAhrensEa` contains one of the dependent model parameters, it will have no effect on the output. Its value in the resulting structure will just be the value specified by the definition given in Table B.2, which can be rewritten using solely independent parameters. Of course, it is possible to generate, or manipulate, a model parameter structure directly, but in doing so, one risks that the dependent model parameter values are inconsistent with the independent parameter values. Thus, it is advised to set the model parameters using the function `setAhrensEa`.

The function `setAhrensEa` uses the auxiliary function `setparam`, which sets specified fields of a given structure to specified values. In the process, it checks whether the specified fields exist in the structure, and it gives an error if that is not the case. Since the function `setparam` is also used to set parameters in the numerical implementation of the models of Kilicarslan (see Appendix A) and

Parameter	Fieldname	Default value	Unit
A_b	`Ab`	8.11×10^{-3}	m^2
A_{tp}	`Atp`	7.92×10^{-4}	m^2
$A_{v,g}$	`Avg`	6.26×10^{-5}	m^2
$C_{D,g}$	`CDg`	0.6	–
c_p	`cp`	1009.7	J/kg/K
ΔH_f	`DHf`	50.016×10^6	J/kg
L_{tp}	`Ltp`	1.52	m
MW_g	`MWg`	16.043	kg/kmol
P_0	`P0`	1.01325×10^5	Pa
r	`r`	19.8	–
R_a	`Ra`	287	J/kg/K
R_u	`Ru`	8.31441×10^3	J/kmol/K
T_0	`T0`	300	K
V_{cc}	`Vcc`	1.97×10^{-3}	m^3

Table B.1: Independent model parameters of the model of Ahrens et al., the fieldnames under which they are stored in the parameter structure, their default values, and the units in which they are expressed.

Parameter	Fieldname	Defined by	Unit
γ	`gam`	$= \frac{c_p}{c_p - R_a}$	–
h_e	`he`	$= h_r + \frac{\Delta H_f}{1+r}$	J/kg
h_r	`hr`	$= c_p T_0$	J/kg
k_g	`kg`	$= \sqrt{2\rho_g} C_{D,g} A_{v,g}$	$\mathrm{kg}^{1/2}\,\mathrm{m}^{1/2}$
$p_{\max}$	`pmax`	$= \frac{1}{0.75} \frac{P_0(1+r)h_r}{\Delta H_f}$	Pa
R_g	`Rg`	$= R_u / \mathrm{MW}_g$	J/kg/K
ρ_g	`rhog`	$= \frac{P_0}{R_g T_0}$	$\mathrm{kg/m}^3$
U_f	`Uf`	$= \sqrt{0.1875} \frac{R T_0 (1+r)^{3/2}}{A_b \sqrt{P_0}} \sqrt{\frac{h_r}{\Delta H_f}}$	m/s

Table B.2: Dependent model parameters of the model of Ahrens et al., the fieldnames under which they are stored in the parameter structure, expressions defining their values, and the units in which they are expressed.

Function	Input	Description
`odeAhrensEa`	`t,y,s`	Returns the value of the function $\mathbf{f}(t, \mathbf{y})$ of (B.4), for specified time `t`, vector `y`, and the model parameters stored in structure `s`.
`setAhrensEa`	none	Returns a parameter structure with *default* values, that can be passed to the function `odeAhrensEa`.
	`'par1',val1,...`	Returns a parameter structure with the *independent* parameters `par`n set to the specified values `val`n; the other parameters have default values. The structure can be passed to the function `odeAhrensEa`.
`setparam`	`s,'fld1',val1,...`	See Appendix D.

Table B.3: Overview of the functions that are defined in Section B.1.

Richards et al. (see Appendix C), its implementation and usage is discussed in Appendix D.

The function `odeAhrensEa` is an implementation of the function $\mathbf{f}(t, \mathbf{y})$ of (B.4). Besides the time t and the vector $\mathbf{y}$, the implemented function also needs a parameter structure specifying the model parameters as input. The code of the function `odeAhrensEa` is given in Listing B.1.2 at the end of this section.

An overview of the MATLAB functions defined in this section, is presented in Table B.3.

Listing B.1.1: Function setAhrensEa. Given a paired list of fieldnames and values, it returns a structure of model parameters, which can be passed to the function `odeAhrensEa`. Model parameter values that are not specified, are set to their default values.

setAhrensEa.m

```
function sOut = setAhrensEa(varargin);

% Set default values

s = struct(...
    'Ab'  , 8.11e-3    ,... % [ m^2 ]
    'Atp' , 7.92e-4    ,... % [ m^2 ]
    'Avg' , 6.26e-5    ,... % [ m^2 ]
    'CDg' , 0.6        ,... % [ - ]
    'cp'  , 1009.7     ,... % [ J/kg/K ]
    'DHf' , 50.016e+6  ,... % [ J/kg ]
    'gam' , []         ,... % (calculated, see below)
    'he'  , []         ,... % (calculated, see below)
    'hr'  , []         ,... % (calculated, see below)
    'kg'  , []         ,... % (calculated, see below)
    'Ltp' , 1.52       ,... % [ m ]
    'MWg' , 16.043     ,... % [ kg/kmol ]
    'P0'  , 1.01325e+5 ,... % [ Pa ]
    'pmax', []         ,... % (calculated, see below)
    'r'   , 19.8       ,... % [ - ]
    'Ra'  , 287        ,... % [ J/kg/K ]
    'Rg'  , []         ,... % (calculated, see below)
    'rhog', []         ,... % (calculated, see below)
    'Ru'  , 8.31441e+3 ,... % [ J/kmol/K ]
    'T0'  , 300        ,... % [ K ]
    'Uf'  , []         ,... % (calculated, see below)
    'Vcc' , 1.97e-3    );   % [ m^3 ]

% Set user specified model parameters

if nargin > 0
    s = setparam(s,varargin{:});
end

% Calculate dependent model parameters

s.gam = s.cp/(s.cp - s.Ra);      % [ - ]
s.hr  = s.cp*s.T0;               % [ J/kg ]
s.he  = s.hr + s.DHf/(1 + s.r);  % [ J/kg ]
s.Rg  = s.Ru/s.MWg;              % [ J/kg/K ]

s.rhog = s.P0/(s.Rg*s.T0);           % [ kg/m^3 ]
s.kg   = sqrt(2*s.rhog)*s.CDg*s.Avg; % [ kg^0.5 m^0.5 ]

s.pmax = s.hr*(1 + s.r)*s.P0/s.DHf/0.75;  % [ Pa ]
s.Uf   = sqrt(0.1875*s.hr/s.DHf/s.P0)*s.kg*(1 + s.r)^1.5*s.Ra* ...
    s.T0/s.Ab;  % [ m/s ]

% Assign output structure

sOut = s;
```

setAhrensEa.m

Listing B.1.2: Function odeAhrensEa. Given the following input arguments: time t, the vector $\mathbf{y}$ (see (B.3)), and a model parameter structure, it returns the vector $\mathbf{f}(t, \mathbf{y})$ defined by (B.4).

odeAhrensEa.m

```
function dydt = odeAhrensEa(t,y,s);

% Compute auxiliary constants

tmp = (s.gam - 1)/s.Vcc;

A = s.kg*(1 + s.r)*tmp*s.hr;
B = s.Ab*s.Uf/s.Ra/s.T0*tmp*s.DHf/(1 + s.r);
C = tmp*s.he;
D = s.Atp/s.Ltp;

```

```
F = @(p) sqrt(abs(min(p,0)));  % function of gauge pressure, p

% Compute dydt
%    y(1) == gauge pressure, [ Pa ]
%    y(2) == mass flux at chamber exit, [ kg/s ]

dy1dt = A*F(y(1)) + B*(y(1) + s.P0)- C*y(2);
dy2dt = D*y(1);

dydt  = [dy1dt; dy2dt];
```

odeAhrensEa.m

B.2 Generating data for graphs

The script file `dataAhrensEa.m` (see Listing B.2.1) contains the code that was used to generate the data for the figures of Section 4.6. By setting the Matlab variable `identifier` in Line 17 to a certain string and running the script, data for the figure associated with the identifier are generated, and stored under an appropriate name. The available choices for the identifier are listed in Table B.4, together with descriptions of the data that are generated for those choices and the filenames under which the data are stored. The data that are stored in the `.mat` files are: the model parameter structure, the parameter δ (needed to specify the initial conditions, see below), the initial values vector, and the solution structure that was returned by the ODE solver.

The initial conditions that were used, are given in Table B.5. The constants B and C (see (B.6) and (B.7)) in specifying the initial conditions conceal the dependency on the model parameters somewhat. To keep the code in the script file `dataAhrensEa.m` clear, the function `iniOdeAhrensEa` was written to produce the initial conditions. It needs the model parameter structure and the 'fine tuning' parameter δ as input arguments. The code of the function `iniOdeAhrensEa` is given in Listing B.2.2.

For completeness, an overview of the Matlab functions defined in this section, is presented in Table B.6.

Listing B.2.1: Script file dataAhrensEa.m. Generates data for the figures of Section 4.6. The choice of the identifier set in Line 17 determines the data that are generated.

dataAhrensEa.m

```
 1  % dataAhrensEa.m
 2
 3  % Clear workspace, close figure windows, and clear command window
 4
 5  clear all
 6  close all
 7  clc
 8
 9  % Set identifier for generation of data for a certain figure from
10  % the Interim Report, Chapter 4. Choose from:
11  %
```

Identifier	Filename	Description
`'Figure4_3a'`	`dataFigure4_3a.mat`	Generates data for Figure 4.3a. Default parameter values are used, $\delta = 0.242$.
`'Figure4_3b'`	`dataFigure4_3b.mat`	Generates data for Figure 4.3b. Default parameter values are used, $\delta = 0.243$.
`'Figure4_4'`	`dataFigure4_3a.mat` `dataFigure4_3b.mat`	Generates data for Figure 4.4. Same as running the script with the identifier choices `'Figure4_3a'` and `'Figure4_3b'` consecutively.

Table B.4: Identifier choices for the script file `dataAhrensEa.m` (see Line 17 of Listing B.2.1), and the filenames under which the generated data are stored. The following data are stored in the `.mat` files: the parameter structure (`s`), the parameter δ (`delta`), the initial values vector (`y0`), and the solution structure (`sol`) that is returned by the ODE solver.

Initial condition		Value used	Unit
`y0(1)`	$= p(0) = p_0$	$(1+\delta)p_{\max}$	Pa
`y0(2)`	$= \dot{m}_e(0) = \dot{m}_{e,0}$	$\frac{B}{C}(p_0 + P_0)$	kg/s

Table B.5: Initial values, stored in the vector `y0` that was passed to the ODE solver to generate the data for the figures of Section 4.6, expressed in model parameters. In the code, they can be obtained with the function `iniOdeAhrensEa` (see Listing B.2.2), which needs the parameter structure and the parameter δ as input arguments.

Function	Input	Description
`iniOdeAhrensEa`	`s,delta`	Returns the vector of initial values `y0` as specified in Table B.5, which depend on the model parameters (stored in structure `s`) and the parameter δ (`delta`).

Table B.6: Overview of the functions that are defined in Section B.2.

```
%   'Figure4_3a'  = default parameter values, delta = 0.242
%   'Figure4_3b'  = default parameter values, delta = 0.243
%   'Figure4_4'   -> produces same data as 'Figure4_3a' and
%                     'Figure4_3b' separately

identifier = 'Figure4_3';

% Set options for ODE-solver

options = odeset('AbsTol',1e-8,'RelTol',1e-6);

% Set time-span for ODE-solver

tmin  = 0;     % [ s ]
tmax  = 0.18;  % [ s ]
tspan = [tmin tmax];

% Initialize structure of model parameters (default values)

s = setAhrensEa;

if ~strcmp(identifier,'Figure4_3a') && ...
        ~strcmp(identifier,'Figure4_3b') && ...
        ~strcmp(identifier,'Figure4_4')
    error('Unknown identifier choice.');
end

% Select data to generate for plotting the figure, and save it

if strcmp(identifier,'Figure4_3a') || strcmp(identifier,'Figure4_4')

    % Set initial values for ODE-solver for default values with
    % delta = 0.242
    %    y0(1) == initial gauge pressure, [ Pa ]
    %    y0(2) == initial mass flux at chamber exit, [ kg/s ]

    delta = 0.242;
    y0 = iniOdeAhrensEa(s,delta);

    % Run ODE-solver, and save

    sol = ode45(@odeAhrensEa,tspan,y0,options,s);
    save('dataFigure4_3a','s','delta','y0','sol')

end

if strcmp(identifier,'Figure4_3b') || strcmp(identifier,'Figure4_4')

    % Set initial values for ODE-solver for default values with
    % delta = 0.243
    %    y0(1) == initial gauge pressure, [ Pa ]
    %    y0(2) == initial mass flux at chamber exit, [ kg/s ]

    delta = 0.243;
    y0 = iniOdeAhrensEa(s,delta);

    % Run ODE-solver, and save

    sol = ode45(@odeAhrensEa,tspan,y0,options,s);
    save('dataFigure4_3b','s','delta','y0','sol')

end
```

dataAhrensEa.m

Listing B.2.2: Function iniOdeAhrensEa. Given a model parameter structure (`s`) and the parameter δ (`delta`), the function returns the vector of initial values $(p_0, \dot{m}_{e,0})^T$ specified in Table B.5.

iniOdeAhrensEa.m

```
function y0 = iniOdeAhrensEa(s,delta);

% Compute auxiliary constants B and C (needed for specifying initial
% mass flux)

tmp = (s.gam - 1)/s.Vcc;

B = s.Ab*s.Uf/s.Ra/s.T0*tmp*s.DHf/(1 + s.r);
C = tmp*s.he;

% Set initial values for ODE-solver
%    y0(1) == initial gauge pressure, [ Pa ]
%    y0(2) == initial mass flux at chamber exit, [ kg/s ]

y01 = (1 + delta)*s.pmax;
y02 = B*(y01 + s.P0)/C;

y0 = [y01; y02];
```

iniOdeAhrensEa.m

B.3 Plotting graphs

The script file `plotAhrensEa.m` (see Listing B.3.1) contains the code that was used to plot the figures of Section 4.6, using the data generated with the script `dataAhrensEa` (see Section B.2). By setting the MATLAB variable `identifier` in Line 16 to a certain string and running the script, data for the figure associated with that identifier is loaded, and a graph is plotted and saved under an appropriate name. The available choices for the identifier are listed in Table B.7. It mentions which figures are produced for those choices, and the filenames under which the figures are saved (i.e. the `.fig` files). The data files (extension: `.mat`) that are needed to plot the figures, are shown between angular brackets.

In Figure 4.4, the graph of the time-derivative of the gauge pressure ($\frac{dp}{dt}$) is plotted versus the gauge pressure (p). The time-derivative was not computed within the script `dataAhrensEa`, so it must be computed within the script `plotAhrensEa`. To keep the code of the script clear, the function `derivative`

Identifier	Filename ⟨needed⟩	Description
`'Figure4_3a'`	`Figure4_3a.fig` ⟨`dataFigure4_3a.mat`⟩	Plots Figure 4.3a.
`'Figure4_3b'`	`Figure4_3b.fig` ⟨`dataFigure4_3b.mat`⟩	Plots Figure 4.3b.
`'Figure4_4'`	`Figure4_4.fig` ⟨`dataFigure4_3a.mat`⟩ ⟨`dataFigure4_3b.mat`⟩	Plots Figure 4.4.

Table B.7: Identifier choices for the script file `plotAhrensEa.m` (see Line 16 of Listing B.3.1). The figures are saved as `.fig` files. Data that is needed to plot the figures, is loaded from the `.mat` files; they are indicated by angular brackets.

was written, and included as a subfunction in the script file `plotAhrensEa.m` (see Listing B.3.1, Lines 138ff.). Given the following input arguments: gauge pressure p (`p`), fuel mass flux at the combustion chamber exit $\dot{m}_e$ (`medot`), and a model parameter structure (`s`), it returns the time-derivative of the gauge pressure $\frac{dp}{dt}$ as given by $f_1(t, \mathbf{y})$ in equation (B.4).

Listing B.3.1: Script file plotAhrensEa.m. Plots the figures from Section 4.6, using data generated with `dataAhrensEa.m` (see Section B.2). The choice of the identifier set in Line 16 determines the graph that is plotted.

plotAhrensEa.m

```
1   % plotAhrensEa.m
2
3   % Clear workspace, close figure windows, and clear command window
4
5   clear all
6   close all
7   clc
8
9   % Set identifier to plot a certain graph from the Interim Report,
10  % Chapter 4. Choose from:
11  %
12  %   'Figure4_3a'  = plot Figure 4.3a from Interim Report
13  %   'Figure4_3b'  = plot Figure 4.3b from Interim Report
14  %   'Figure4_4'   = plot Figure 4.4  from Interim Report
15
16  identifier = 'Figure4_3';
17
18  % Select data to load, plot requested figure, and save it
19
20  switch identifier
21
22      case 'Figure4_3a'
23
24          % Load data
25
26          load dataFigure4_3a
27
```

```
        % Extract data for time and gauge pressure from solution
        % structure

        t = sol.x.';
        y = sol.y.';
        p = y(:,1);      % gauge pressure, [ Pa ]

        % Plot graph, annotate, and save

        h = figure('Name','Figure 4.3(a) of Interim Report');
        plot(t,p*1e-3)  % time in [ s ], gauge pressure in [ kPa ]

        title('Figure 4.3(a), delta = 0.242')
        xlabel('Time (s)')
        ylabel('Gauge pressure (kPa)')
        xlim([0 max(t)])
        ylim([-s.P0 s.P0]/500)
        grid on

        saveas(h,'Figure4_3a','fig')

    case 'Figure4_3b'

        % Load data

        load dataFigure4_3b

        % Extract data for time and gauge pressure from solution
        % structure

        t = sol.x.';
        y = sol.y.';
        p = y(:,1);      % gauge pressure, [ Pa ]

        % Plot graph, annotate, and save

        h = figure('Name','Figure 4.3(b) of Interim Report');
        plot(t,p*1e-3)  % time in [ s ], gauge pressure in [ kPa ]

        title('Figure 4.3(b), delta = 0.243')
        xlabel('Time (s)')
        ylabel('Gauge pressure (kPa)')
        xlim([0 max(t)])
        ylim([-s.P0 s.P0]/500)
        grid on

        saveas(h,'Figure4_3b','fig')

    case 'Figure4_4'

        % Load data for delta = 0.242

        load dataFigure4_3a

        % Extract data for time, gauge pressure, and mass flux at
        % combustion chamber exit

        t = sol.x.';
        y = sol.y.';
        p = y(:,1);      % gauge pressure, [ Pa ]
        medot = y(:,2);  % mass flux at chamber exit, [ kg/s ]

        % Compute time-derivative of gauge pressure from ODEs

        dpdt = derivative(p,medot,s);

        % Plot graph

        h = figure('Name','Figure 4.4 of Interim Report');
        plot(p*1e-3,dpdt*1e-3)  % gauge pressure in [ kPa ],
                                % time in [ s ]

        % Load data for delta = 0.243

        load dataFigure4_3b

        % Extract data for time, gauge pressure, and mass flux at
        % combustion chamber exit

        t = sol.x.';
        y = sol.y.';
        p = y(:,1);      % gauge pressure, [ Pa ]
        medot = y(:,2);  % mass flux at chamber exit, [ kg/s ]

        % Compute time-derivative of gauge pressure from ODEs

        dpdt = derivative(p,medot,s);

        % Add to previous graph, annotate, and save

        hold on
        plot(p*1e-3,dpdt*1e-3)  % gauge pressure in [ kPa ], its
                                % time-derivative in [ kPa/s ]

        title('Figure 4.4')
        xlabel('Gauge pressure p (kPa)')
        ylabel('dp/dt (kPa/s)')
        axis([-100 100 -5e4 5e4])
        grid on

        saveas(h,'Figure4_4','fig')

    otherwise

        error('Unknown identifier choice.');

end  % switch

```

```
%-------------------------------------------------------------------

function dpdt = derivative(p,medot,s);

   % Compute auxiliary constants

   tmp = (s.gam - 1)/s.Vcc;

   A = s.kg*(1 + s.r)*tmp*s.hr;
   B = s.Ab*s.Uf/s.Ra/s.T0*tmp*s.DHf/(1 + s.r);
   C = tmp*s.he;

   F = @(p) sqrt(abs(min(p,0)));  % function of gauge pressure, p

   % Compute dpdt, [ Pa/s ]

   dpdt = A*F(p) + B*(p + s.P0)- C*medot;
```

plotAhrensEa.m

Appendix C

MATLAB codes for model of Richards et al.

In this chapter, the system of ordinary differential equations (ODEs) of the model of Richards et al. (see (5.73)–(5.89)) is numerically implemented using MATLAB [30] software. The standard MATLAB ODE solver `ode45` is used to solve the equations. It is assumed that the user is familiar with MATLAB, and the ODE solver.

The MATLAB ODE solvers for initial value problems (a.o. the solver `ode45`) can be used to solve systems of ODEs written in the general form

$$\frac{d\mathbf{y}}{dt} = \mathbf{f}(t, \mathbf{y}), \tag{C.1}$$

$$\mathbf{y}(0) = \mathbf{y}_0, \tag{C.2}$$

where $\mathbf{y}$ is a vector, $\mathbf{f}$ is a vector function, t is time, and $\mathbf{y}_0$ is a vector of initial values. To obtain a numerical solution, a function implementing the right-hand side of (C.1) must be passed to the solver, together with the initial values vector $\mathbf{y}_0$. The function that is passed to the solver, should return a column vector.

In Section C.1 the model equations of Richards et al. are written in the general form of (C.1) and (C.2), and the function that can be passed to the solver is given. In Section C.2 the code is given that was used to generate the data for the figures of Section 5.4. Finally, Section C.4 presents the code that was used to plot the figures of Section 5.4, using the generated data.

It should be mentioned that the code was written rather quickly, and no attempt has been made to optimize it. In the implementation of the functions, no attention is paid to checking whether the input arguments are of the correct form. Thus, the user should take care when prescribing the input arguments, although eventually an error will be produced if the input is of an unexpected form. Also, plots produced by the code of Section C.4 may differ somewhat from the figures in Section 5.4. This is because they were manipulated manually (e.g. by resizing them, replacing labels) before including them in the report.

C.1 Numerical implementation

The model equations of Richards et al. (see (5.73)–(5.89), (5.97)), can be written in the general form of (C.1) and (C.2) by defining $\mathbf{y}$ and $\mathbf{f}$ in the following way:

$$\mathbf{y} = \begin{pmatrix} \widetilde{P} \\ \widetilde{T} \\ \widetilde{u} \\ Y_f \end{pmatrix}, \tag{C.3}$$

and, writing $\widetilde{P}$, $\widetilde{T}$, $\widetilde{u}$, Y_f instead of y_1, y_2, y_3, y_4, respectively, (for easier reference to the model equations)

$$\mathbf{f}(t,\mathbf{y}) = \begin{pmatrix} \gamma\left(A + B\cdot RR + CD - (C + G\widetilde{Z}_e)\,\widetilde{T}\right) \\ \left(f_1(t,\mathbf{y}) - (A - G\widetilde{Z}_e)\widetilde{T}\right)\dfrac{\widetilde{T}}{\widetilde{P}} \\ E(\widetilde{P}_e - 1)\,\dfrac{\widetilde{T}_e}{\widetilde{P}_e} - F\,\widetilde{u}\,|\widetilde{u}| \\ \left(A(Y_{f,i} - Y_f) - RR\right)\dfrac{\widetilde{T}}{\widetilde{P}} \end{pmatrix}, \tag{C.4}$$

where RR, $\widetilde{P}_e$, $\widetilde{T}_e$ and $\widetilde{Z}_e$ are functions of $\widetilde{P}$, $\widetilde{T}$, $\widetilde{u}$ and Y_f given by

$$RR = A'\,\frac{\widetilde{P}^2}{\widetilde{T}^{3/2}}\,Y_f^2\exp(-\widetilde{T}_{\mathrm{act}}/\widetilde{T}), \tag{C.5}$$

$$\widetilde{T}_e = \widetilde{T}\left(1 - \frac{\gamma-1}{2}\,\frac{u_0^2}{\gamma R T_0}\,\frac{\widetilde{u}^2}{\widetilde{T}}\right), \tag{C.6}$$

$$\widetilde{P}_e = \widetilde{P}\left(\frac{\widetilde{T}_e}{\widetilde{T}}\right)^{\frac{\gamma}{\gamma-1}}, \tag{C.7}$$

$$\widetilde{Z}_e = \frac{\widetilde{P}_e}{\widetilde{T}_e}\,\widetilde{u} \tag{C.8}$$

and A, B, C, D, E, F and G are expressed in the model parameters by:

$$A = \frac{1}{\tau_f} = \frac{\dot{m}_i}{\rho_0 V_{cc}}, \tag{C.9}$$

$$B = \frac{\Delta H_f}{c_p T_0}, \tag{C.10}$$

$$C = \frac{\widehat{h}}{L_1 c_p \rho_0}, \tag{C.11}$$

$$D = \frac{T_w}{T_0}, \tag{C.12}$$

$$E = \frac{R T_0}{L_{tp} u_0}, \tag{C.13}$$

$$F = \frac{u_0 f}{2 D_{tp}}, \tag{C.14}$$

$$G = \frac{u_0}{L_2}. \tag{C.15}$$

Parameter	Fieldname	Default value	Unit
A'	`Aprime`	3.85×10^8	1/s
c_p	`cp`	1350	J/kg/K
ΔH_f	`DHf`	5×10^7	J/kg
D_{tp}	`Dtp`	0.0178	m
f	`f`	0.03	–
h	`h`	120	W/m^2/K
$1/\tau_f$	`itauf`	1/0.030	1/s
L_1	`L1`	0.0119	m
L_2	`L2`	0.8486	m
L_{tp}	`Ltp`	0.6100	m
P_0	`P0`	1.01325×10^5	Pa
R	`R`	287	J/kg/K
T_0	`T0`	300	K
T_{act}	`Tact`	50	–
τ_m	`taum`	0	s
T_w	`Tw`	1000	K
$Y_{f,i}$	`Yfi`	0.06	–

Table C.1: Independent model parameters of the model of Richards et al., the fieldnames under which they are stored in the parameter structure, their default values, and the units in which they are expressed.

Note that $f_2(t, \mathbf{y})$ is defined using $f_1(t, \mathbf{y})$; it makes the notation more compact. In fact, the numerical implementation of the vector function $\mathbf{f}$ of (C.4) is done in an analogous way. To incorporate the mixing time τ_m in the model equations, the term RR in the expression for f_1 should be replace by $\widehat{RR}$ defined by $\widehat{RR} = \frac{RR}{1+B\cdot RR\tau_m}$ (see (5.97)).

The function that must be passed to the solver (implementing $\mathbf{f}(t, \mathbf{y})$), depends on the model parameters. To be able to vary all of the model parameters, it was chosen to pass a structure of model parameters to that function. The structure has fields in which the model parameter values are stored; the fieldnames correspond to the model parameters. Such a structure can be generated with help of the function `setRichardsEa` (see Listing C.1.1 at the end of this section). Calling the function without any input arguments returns a structure with the model parameters set to their default values. Calling the function with input of the form `'par1',val1,...` (i.e. in pairs of fieldnames and values), sets the model parameters specified by the fieldnames `par`n to the corresponding values `val`n, while the other parameters are set to their default values. Together, Tables C.1 and C.2 list the model parameters that are stored in a parameter structure, their fieldnames, and the default values.

Not all model parameters can be specified independently: those that can, are listed in Table C.1, while those that depend on others are listed in Table C.2. While no error is produced if the input of the function `setRichardsEa` contains one of the dependent model parameters, it will have no effect on the output. Its value in the resulting structure will just be the value specified by the definition given in Table C.2, which can be rewritten using solely independent parameters. Of course, it is possible to generate, or manipulate, a model parameter structure

Parameter	Fieldname	Defined by	Unit
γ	`gam`	$= \frac{c_p}{c_p - R}$	–
ρ_0	`rho0`	$= \frac{P_0}{RT_0}$	kg/m^3
u_0	`u0`	$= \sqrt{\gamma R T_0}$	m/s

Table C.2: Dependent model parameters of the model of Richards et al., the fieldnames under which they are stored in the parameter structure, expressions defining their values, and the units in which they are expressed.

Function	Input	Description
`odeRichardsEa`	`t,y,s`	Returns the value of the function $\mathbf{f}(t, \mathbf{y})$ of (C.4), for specified time `t`, vector `y`, and the model parameters stored in structure `s`.
`setparam`	`s,'fld1',val1,...`	See Appendix D.
`setRichardsEa`	none	Returns a parameter structure with *default* values, that can be passed to the function `odeRichardsEa`.
	`'par1',val1,...`	Returns a parameter structure with the *independent* parameters `par`n set to the specified values `val`n; the other parameters have default values. The structure can be passed to the function `odeRichardsEa`.

Table C.3: Overview of the functions that are defined in Section C.1.

directly. As long as none of the model parameters γ, ρ_0, or u_0 are changed, there is no objection against manipulating the structure of model parameters without using the function `setRichardsEa`. However, care should be taken to spell the fieldnames of the structure correctly, because if an incorrect name is used, a field will be added to the structure (instead of changing the contents of the intended field). No warning will be given in such a case, while if a similar mistake is made using the function `setRichardsEa`, an error will be given.

The function `setRichardsEa` uses the auxiliary function `setparam`, which sets specified fields of a given structure to specified values. In the process, it checks whether the specified fields exist in the structure, and it gives an error if that is not the case. Since the function `setparam` is also used to set parameters in the numerical implementation of the models of Kilicarslan (see Appendix A) and Ahrens et al. (see Appendix B), its implementation and usage is discussed in Appendix D.

The function `odeRichardsEa` is an implementation of the function $\mathbf{f}(t, \mathbf{y})$ in (C.4). Besides the time t and the vector $\mathbf{y}$, the implemented function also needs a parameter structure specifying the model parameters as input. The code of the function `odeRichardsEa` is given in Listing C.1.2 at the end of this section.

An overview of the MATLAB functions defined in this section, is presented in Table C.3.

Listing C.1.1: Function setRichardsEa. Given a paired list of fieldnames and values, it returns a structure of model parameters, which can be passed to the function `odeRichardsEa`. Model parameter values that are not specified, are set to their default values.

setRichardsEa.m

```
function sOut = setRichardsEa(varargin);

% Set default values

s = struct(...
    'Aprime', 3.85e+8    ,... % [ 1/s ]
    'cp'    , 1350       ,... % [ J/kg/K ]
    'DHf'   , 5e+7       ,... % [ J/kg ]
    'Dtp'   , 0.0178     ,... % [ m ]
    'f'     , 0.03       ,... % [ - ]
    'gam'   , []         ,... % (calculated, see below)
    'h'     , 120        ,... % [ W/m^2/K ]
    'itauf' , 1/0.030    ,... % [ 1/s ] (inverse of tauf)
    'L1'    , 0.0119     ,... % [ m ]
    'L2'    , 0.8486     ,... % [ m ]
    'Ltp'   , 0.6100     ,... % [ m ]
    'P0'    , 1.01325e+5 ,... % [ Pa ]
    'R'     , 287        ,... % [ J/kg/K ]
    'rho0'  , []         ,... % (calculated, see below)
    'T0'    , 300        ,... % [ K ]
    'Tact'  , 50         ,... % [ - ]
    'taum'  , 0          ,... % [ s ]
    'Tw'    , 1000       ,... % [ K ]
    'u0'    , []         ,... % (calculated, see below)
    'Yfi'   , 0.06      );   % [ - ]

% Set user specified model parameters

if nargin > 0
    s = setparam(s,varargin{:});
end

% Calculate dependent model parameters

s.gam  = s.cp/(s.cp - s.R);     % [ - ]
s.rho0 = s.P0/(s.R*s.T0);       % [ kg/m^3 ]
s.u0   = sqrt(s.gam*s.R*s.T0);  % [ m/s ]

% Assign output structure

sOut = s;
```
setRichardsEa.m

Listing C.1.2: Function odeRichardsEa. Given the following input arguments: time t, the vector $\mathbf{y}$ (see (C.3)), and a model parameter structure, it returns the vector $\mathbf{f}(t, \mathbf{y})$ defined by (C.4).

odeRichardsEa.m
```
function dydt = odeRichardsEa(t,y,s);

% Compute auxiliary constants

A = s.itauf;
B = s.DHf/(s.cp*s.T0);
C = s.h/(s.L1*s.cp*s.rho0);
D = s.Tw/s.T0;
E = s.R*s.T0/(s.Ltp*s.u0);
F = 0.5*s.u0*s.f/s.Dtp;
G = s.u0/s.L2;

% Compute time dependent variables

P  = y(1);
T  = y(2);
u  = y(3);
Yf = y(4);

Te = T*(1 - 0.5*(s.gam - 1)*s.u0^2*u^2/(s.gam*s.R*s.T0*T));
Pe = P*(Te/T)^(s.gam/(s.gam - 1));
Ze = u*Pe/Te;
RR = s.Aprime*(P^2/T^1.5)*Yf^2*exp(-s.Tact/T);
RR = RR/(1 + B*RR*s.taum);  % Reaction rate with mixing incorporated

% Compute dydt

dPdt  = s.gam*(A + B*RR + C*D) - s.gam*(G*Ze + C)*T;
```

```
dTdt  = (dPdt - (A - G*Ze)*T)*T/P;
dudt  = E*(Pe - 1)*Te/Pe - F*u*abs(u);
dYfdt = (A*(s.Yfi - Yf) - RR)*T/P;

dydt  = [dPdt; dTdt; dudt; dYfdt];
```
odeRichardsEa.m

C.2 Generating data for graphs

The script file `dataRichardsEa.m` (see Listing C.2.1) contains the code that was used to generate the data for the figures of Section 5.4. By setting the MATLAB variable `identifier` in Line 28 to a certain string and running the script, data for the figure associated with the identifier are generated, and stored under an appropriate name. The available choices for the identifier are listed in Table C.4, together with descriptions of the data that are generated for those choices and the filenames under which the data are stored. The initial conditions that were used, are given in Table C.5. They are the same for all choices of the identifier. For uniformity in the generated data, the time span that was passed to the ODE solver was also the same for all identifier choices, namely 0–1.3 s. Although this is somewhat larger than is necessary to plot some of the figures, it is the interval that is needed for generating nearly all of the data used for plotting the figures of Section 5.4.

For each of the identifier choices, one or two parameter values are varied over a certain range, using a fixed step size. The parameter values are stored in an array named ⟨*par*⟩`Data`, where ⟨*par*⟩ is the name of the parameter that is varied. For each of the parameter values in the array, a corresponding solution structure can be obtained from the ODE solver. These solution structures are stored in a cell array with the name `sol`⟨*par*⟩, where ⟨*par*⟩ is the name of the parameter that is varied. The parameters that are varied, the names of the arrays in which the various values are stored, and the names of the cell arrays with the corresponding solution structures are given in Table C.6 for each of the identifier choices.

Before running the script, a word of caution is in order:

> **Warning:** Running the script `dataRichardsEa` with one of the choices `'Figure5_10'`, `'Figure5_11'`, or `'Figure5_12'` for the variable `identifier` (see Line 28 of Listing C.2.1) may take a considerable time to finish! Also, very large data structures are generated, so one should have a large amount of memory and storage space available! See Table C.7 for an indication of how much time and memory/storage space is needed.

As can be seen from Table C.7, the amount of data generated is larger for smaller values of the mixing time τ_m. The reason for this is, that for smaller mixing times the model equations have oscillatory solutions for a wider range of the model parameters τ_f and T_w, as can be seen from Figures 5.10–5.12. Since the ODE solver can use a larger step size if the solution does not change very rapidly, the solution structures that are returned by the ODE solver, are much smaller if flame extinction or steady combustion occurs than for an oscillatory combustion process. The amounts of data that are generated for the three choices of the

Identifier	Filename	Description
`'Figure5_3'` `'Figure5_4'` `'Figure5_5'`	`dataFigures5_3-4-5.mat`	Generates data for Figures 5.3–5.5.
`'Figure5_6'`	`dataFigure5_6.mat`	Generates data for Figure 5.6.
`'Figure5_7'` `'Figure5_8'`	`dataFigures5_7-8.mat`	Generates data for Figures 5.7–5.8.
`'Figure5_9'`	`dataFigure5_9.mat`	Generates data for Figure 5.9.
`'Figure5_10'`	`dataFigure5_10.mat`	Generates data for Figure 5.10.
`'Figure5_11'`	`dataFigure5_11.mat`	Generates data for Figure 5.11.
`'Figure5_12'`	`dataFigure5_12.mat`	Generates data for Figure 5.12.

Table C.4: Identifier choices for the script file `dataRichardsEa.m` (see Line 28 of Listing C.2.1), and the filenames under which the generated data are stored. The data that are saved in the `.mat` files, are: the parameter structure (`s`), the initial values vector (`y0`), the array of values of the parameter that was varied (see Table C.6) and the corresponding cell array containing the solution structures (see Table C.6).

Initial condition		Value used	Unit
`y0(1)`	$= \widetilde{P}(0)$	1	–
`y0(2)`	$= \widetilde{T}(0)$	5	–
`y0(3)`	$= \widetilde{u}(0)$	0	–
`y0(4)`	$= Y_f(0) = Y_{f,i}$	0.06	–

Table C.5: Initial values, stored in the vector `y0` that was passed to the ODE solver to generate the data for the figures of Section 5.4.

Identifier	Par	Range	Unit	Array	Structure
`'Figure5_3'` `'Figure5_4'` `'Figure5_5'`	T_w	$750, 1000, 1200$	K	`TwData`	`solTw`
`'Figure5_6'`	h	$100 : 5 : 165$	$\mathrm{W/m^2/K}$	`hData`	`solh`
`'Figure5_7'` `'Figure5_8'`	f	$0, 0.02, 0.0325$	$-$	`fData`	`solf`
`'Figure5_9'`	L_{tp}	$0.4 : 0.025 : 0.8$	m	`LtpData`	`solLtp`
`'Figure5_10'`	τ_f	$0.01 : 0.0025 : 0.09$	s	`taufData`	`soltaufTw`
	T_w	$700 : 25 : 1400$	K	`TwData`	
	τ_m	$= 0$	s		
`'Figure5_11'`	τ_f	$0.01 : 0.0025 : 0.09$	s	`taufData`	`soltaufTw`
	T_w	$700 : 25 : 1400$	K	`TwData`	
	τ_m	$= 1.0 \times 10^{-3}$	s		
`'Figure5_12'`	τ_f	$0.01 : 0.0025 : 0.09$	s	`taufData`	`soltaufTw`
	T_w	$700 : 25 : 1400$	K	`TwData`	
	τ_m	$= 0.5 \times 10^{-3}$	s		

Table C.6: The model parameters that are varied to obtain data for the figures of Section 5.4, associated with the identifier choices. The ranges over which the parameters are varied (expressed in the listed units), are given in the format start:step:end, or as a list of values. The parameter values are stored in arrays and the corresponding solution structures in cell arrays; their names are listed in the last two columns.

Identifier	τ_m (ms)	CPU time (min.)	Data (MB)
`'Figure5_10'`	0	21.0	889
`'Figure5_11'`	1.5	10.7	453
`'Figure5_12'`	0.5	15.1	638

Table C.7: CPU time taken and size of data generated by running the script `dataRichardsEa` for the three listed identifier choices (see Line 28 of Listing C.2.1). Each identifier choice is associated with a mixing time τ_m. The CPU times were measured with the MATLAB command `cputime`. The size of the data is the storage space occupied in the workspace after the script has finished. (In storage on hard disk less space is occupied, because MATLAB compresses it by default.) Used computer configuration: desktop with Intel Core 2 Duo-processor E6300 and 1024MB DDR2 SDRAM memory (533MHz, 64-bits), and Windows XP as operating system.

identifier are very large, because the model parameters τ_f and T_w are varied over 33 and 29 values, respectively, resulting in cell arrays that store $33 \times 29 = 957$ solution structures.

Of course, one way of limiting the demand on time and storage space is by varying these parameters over less values. This can be done without any problems by modifying the script file `dataRichardsEa.m` in Lines 127 and 128, where `taufData` and `TwData` are specified. (In fact, all model parameter ranges can be adjusted in an analogous manner, without a need to rewrite other code.) Then, when plotting the figures using the modified data, a graph of lower resolution will result.

The data that were generated with the script `dataRichardsEa` was used to examine the time history of the solutions for many different parameter values. But if the data is only needed to plot the Figures 5.10–5.12 of Section 5.4, it would be better to combine the code of the script file `dataRichardsEa.m` with the code of the script file `freqDataRichardsEa.m`, which is given in the next section. It was written to determine the amplitudes and frequencies of the oscillatory solutions. By determining and storing the amplitude and frequency information directly after a solution structure has been obtained, there is no need to store the solution structures for later use. Then, the demands on memory or storage capacity are minimal. It will probably not make much difference on the needed CPU time.

Listing C.2.1: Script file dataRichardsEa.m. Generates data for the figures of Section 5.4. The choice of the identifier set in Line 28 determines the data that are generated.

dataRichardsEa.m

```
 1  % dataRichardsEa.m
 2  
 3  % Clear workspace, close figure windows, and clear command window
 4  
 5  clear all
 6  close all
 7  clc
 8  
 9  % Set identifier for generation of data for a certain figure from
10  % the Interim Report, Chapter 5. Choose from:
11  %
12  %   'Figure5_3'  = Three wall temperature values
13  %   'Figure5_4'  = -> produces same data as 'figure5_3'
14  %   'Figure5_5'  = -> produces same data as 'figure5_3'
15  %   'Figure5_6'  = Range of heat transfer coefficients
16  %   'Figure5_7'  = Three friction coefficient values
17  %   'Figure5_8'  = -> produces same data as 'figure5_7'
18  %   'Figure5_9'  = Range of tailpipe lengths
19  %   'Figure5_10' = Range of flow times and wall temperatures, ...
20  %                  taum = 0 s
21  %   'Figure5_11' = (idem), taum = 1.0e-3 s
22  %   'Figure5_12' = (idem), taum = 0.5e-3 s
23  %
24  % WARNING! Choices 'Figure5_10', 'Figure5_11', 'Figure5_12' may take
```

```
% a considerable time to process and they produce very large data
% structures!

identifier = 'Figure5_3';

% Set initial values for ODE-solver

Pini  = 1;     % [ - ]
Tini  = 5;     % [ - ]
uini  = 0;     % [ - ]
Yfini = 0.06;  % [ - ]

y0 = [Pini; Tini; uini; Yfini];

% Set options for ODE-solver

options = odeset('AbsTol',1e-8,'RelTol',1e-6);

% Set time-span for ODE-solver

tmin  = 0;    % [ s ]
tmax  = 1.3;  % [ s ]
tspan = [tmin tmax];

% Initialize structure of model parameters

s = setRichardsEa;

% Select data to generate for plotting the figure, and save it

switch identifier

    case {'Figure5_3','Figure5_4','Figure5_5'}

        % Run ODE-solver with default values for three wall
        % temperature values

        TwData = [750, 1000, 1200];  % [ K ]
        for n = 1:length(TwData)
            s.Tw = TwData(n);
            sol = ode45(@odeRichardsEa,tspan,y0,options,s);
            solTw(n) = {sol};
        end
        filename = 'dataFigures5_3-4-5';
        save(filename,'s','y0','TwData','solTw')

    case 'Figure5_6'

        % Run ODE-solver with default values for a range of heat
        % transfer coefficients

        hData = linspace(100,165,14);  % [ W/m^2/K ], step = 5
        for n = 1:length(hData)
            s.h = hData(n);
            sol = ode45(@odeRichardsEa,tspan,y0,options,s);
            solh(n) = {sol};
        end
        filename = ['data',identifier];
        save(filename,'s','y0','hData','solh')

    case {'Figure5_7','Figure5_8'}

        % Run ODE-solver with default values for three friction
        % coefficient values

        fData = [0, 0.02, 0.0325];  % [ - ]
        for n = 1:length(fData)
            s.f = fData(n);
            sol = ode45(@odeRichardsEa,tspan,y0,options,s);
            solf(n) = {sol};
        end
        filename = 'dataFigures5_7-8';
        save(filename,'s','y0','fData','solf')

    case 'Figure5_9'

        % Run ODE-solver with default values for a range of tailpipe
        % lengths

        LtpData = linspace(0.4,0.8,17);  % [ m ], step = 0.025
        for n = 1:length(LtpData)
            s.Ltp = LtpData(n);
            sol = ode45(@odeRichardsEa,tspan,y0,options,s);
            solLtp(n) = {sol};
        end
        filename = ['data',identifier];
        save(filename,'s','y0','LtpData','solLtp')

    case {'Figure5_10','Figure5_11','Figure5_12'}

        switch identifier
            case 'Figure5_10'
                s.taum = 0;       % [ s ]
            case 'Figure5_11'
                s.taum = 1.0e-3;  % [ s ]
            case 'Figure5_12'
                s.taum = 0.5e-3;  % [ s ]
        end

        % Run ODE-solver with default values for a range of flow
        % times and wall temperatures

        taufData = linspace(0.01,0.09,33);  % [ 1/s ], step = 0.0025
        TwData   = linspace(700,1400,29);   % [ K ],   step = 25
        for n = 1:length(taufData)
            s.itauf = 1/taufData(n);
            for m = 1:length(TwData)
                s.Tw = TwData(m);
                sol = ode45(@odeRichardsEa,tspan,y0,options,s);
                soltaufTw(n,m) = {sol};
            end
        end
        filename = ['data',identifier];
        save(filename,'s','y0','taufData','TwData','soltaufTw')

    otherwise

        error('Unknown identifier choice.');

end  % switch
```

dataRichardsEa.m

C.3 Calculating amplitudes and frequencies

The script file `freqDataRichardsEa.m` (see Listing C.3.1) contains the code that was used to extract the amplitude and frequency information for the Figures 5.9–5.12 of Section 5.4 from the data generated with `dataRichardsEa.m` (see Section C.2). By setting the MATLAB variable `identifier` in Line 21 to a certain string and running the script, data for the figure associated with the identifier are generated, and stored under an appropriate name. The available choices for the identifier are listed in Table C.8, together with descriptions of the data that are generated for those choices, the filenames under which the data are stored, and the data files that are needed for execution of the script with the given identifier. The data files that are needed for execution of the script are generated with the script `dataRichardsEa` (see Section C.2).

The frequencies of the numerical solutions that were obtained with the script `dataRichardsEa`, are determined from a Discrete Fourier Transformation, using MATLAB's Fast Fourier Transform algorithm `fft`. They are the frequencies that maximize the approximated periodogram. To obtain the frequency information for the periodogram, the time interval $[0.3, 1.3]$ (in s) is divided into 4096 equally spaced intervals, giving 4096 samples points (the point $t = 1.3\,\mathrm{s}$ is not included). The time interval and the number of sample points were chosen after testing on given combinations of sine functions, and on some of the numerical solutions of the model equations. Although the testing was not done rigorously, it was found that a (much) shorter time interval would be unsuitable, because the time domain truncation disturbs the frequency information of the signal. The number of sample points that was chosen (4096), was found to be sufficient to prevent serious aliasing of the frequencies. Note that it is a power of 2; the `fft` is the most efficient for vector lengths that are powers of 2.

The testing that was done to determine the interval length and the number of sample points, suggested that the (discrete) Fourier analysis would not give the amplitudes of the time signal with satisfactory accuracy. The main cause for lack of accuracy of the amplitudes is the time domain truncation. It was found that increasing the time interval gave better results, but that would mean very long computing times. Therefore, the (peak-to-peak) amplitudes were estimated from the sampled data by taking the difference of the maximum and the minimum of the pressure on the time interval 1–1.3 s.

The amplitude information can be used to distinguish between steady combustion or flame extinction (amplitude nearly zero) and oscillatory combustion. To distinguish between flame extinction and combustion (steady, or oscillatory), the (mean) fuel mass fraction can be used. Therefore, for Figures 5.10–5.12, the mean fuel mass fraction is determined over the same time interval as the pressure amplitudes, i.e. 1–1.3 s.

Before running the script, a word of caution is in order:

> **Warning:** Running the script `freqDataRichardsEa` with one of the choices `'Figure5_10'`, `'Figure5_11'`, or `'Figure5_12'` for the variable `identifier` (see Line 21 of Listing C.3.1) may take a considerable time to finish! See Table C.9 for an indication of how much time is needed.

The differences in the CPU times that the script needs to finish, come from the different amounts of data that needs to be analyzed for the frequency content. The smaller the mixing time τ_m, the larger the amount of data, as was explained in Section C.2.

Listing C.3.1: Script freqDataRichardsEa.m

freqDataRichardsEa.m

```
 1  % freqDataRichardsEa.m
 2
 3  % Clear workspace, close figure windows, and clear command window
 4
 5  clear all
 6  close all
 7  clc
 8
 9  % Set identifier to generate frequency data (and amplitude data) for
10  % one of the Figures 5.9-5.12 from the Interim Report, Chapter 5.
11  % Choose from:
12  %
13  %   'Figure5_9'  = generate frequency data for Figure 5.9
14  %   'Figure5_10' = generate frequency data for Figure 5.10
15  %   'Figure5_11' = generate frequency data for Figure 5.11
16  %   'Figure5_12' = generate frequency data for Figure 5.12
17  %
18  % WARNING! Choices 'Figure5_10', 'Figure5_11', 'Figure5_12' may take
19  % a considerable time to process!
20
21  identifier = 'Figure5_9';
22
23  % Select data to load, plot requested figure, and save it
24
25  switch identifier
26
27      case 'Figure5_9'
28
29          % Load data
30
31          load dataFigure5_9
32
```

Identifier	Filename ⟨ needed ⟩	Description
`'Figure5_9'`	`freqDataFigure5_9.mat` ⟨ `dataFigure5_9.mat` ⟩	Extracts amplitude and frequency information for Figure 5.9.
`'Figure5_10'`	`freqDataFigure5_10.mat` ⟨ `dataFigure5_10.mat` ⟩	Extracts amplitude and frequency information for Figure 5.10.
`'Figure5_11'`	`freqDataFigure5_11.mat` ⟨ `dataFigure5_11.mat` ⟩	Extracts amplitude and frequency information for Figure 5.11.
`'Figure5_12'`	`freqDataFigure5_12.mat` ⟨ `dataFigure5_12.mat` ⟩	Extracts amplitude and frequency information for Figure 5.12.

Table C.8: Identifier choices for the script file `freqDataRichardsEa.m` (see Line 21 of Listing C.3.1), the filenames under which the generated data are stored, and the files that are needed for execution of the script (in angular brackets). The data that are saved in the `.mat` files, are: the arrays with the computed the amplitudes (`Amp`) and frequencies (`Freq`), in addition to the parameter structure (`s`), the initial values vector (`y0`), and the arrays of values of the parameters that were varied (cf. Table C.6 in Section C.2). For the contour plots of Figures 5.10–5.12, the computed mean fuel mass fraction (`YfMean`) is also added to the data files.

Identifier	τ_m (ms)	CPU time (min.)
`'Figure5_10'`	0	9.5
`'Figure5_11'`	1.5	4.7
`'Figure5_12'`	0.5	6.5

Table C.9: CPU time taken by running the script `freqDataRichardsEa` for the three listed identifier choices (see Line 21 of Listing C.3.1). Each identifier choice is associated with a mixing time τ_m. The CPU times were measured with the MATLAB command `cputime`. Used computer configuration: desktop with Intel Core 2 Duo-processor E6300 and 1024MB DDR2 SDRAM memory (533MHz, 64-bits), and Windows XP as operating system.

```
        % Select points in time, in [ s ], which are used for the
        % frequency analysis of the pressure oscillations

        tstart = 0.3;
        tend   = 1.3;
        N = 4096;
        t = linspace(tstart,tend,N+1);   % sample times, N intervals
        t(end) = [];             % do not include t = 1.3 -> N points

        freqRes = 1/(tend - tstart);  % frequency resolution
        freqFFT = (0:N-1)*freqRes;    % frequencies [ Hz ] from fft

        % Pre-allocate space

        nLtp = length(LtpData);
        nt   = length(t);

        Amp   = zeros(nLtp,1);
        Freq  = zeros(nLtp,1);

        FFT_P = zeros(nt,1);
        y     = zeros(nt,4);
        P     = zeros(nt,1);

        % The frequency information of the pressure oscillations
        % is contained in the entries 2:N/2 of the FFT.
        % To determine the pressure amplitudes, only the points in
        % time > 1 sec. are used.

        Ifreq = 2:N/2;
        freqFFT = freqFFT(Ifreq);
        I = find((t > 1),1);  % index of first t > 1

        % Extract data for pressure from solution structure, and
        % determine peak-to-peak amplitudes [ kPa ] and frequencies
        % [ Hz ]

        for i = 1:nLtp
            y(:,:) = deval(t,solLtp{i}).';
            P(:,1) = y(:,1);
            Amp(i) = max(P(I:end)) - min(P(I:end));
            FFT_P(:,1) = fft(P);
            [val,ind]  = max(abs(FFT_P(Ifreq)));  % determine max.
                                                  % in periodogram
            Freq(i) = freqFFT(index);
        end
        Amp = Amp*s.P0*1e-3;  % pressure amplitude [ kPa ]

        % Save frequency and amplitude data

        filename = ['freqData',identifier];
        save(filename,'s','y0','LtpData','Amp','Freq')

    case {'Figure5_10','Figure5_11','Figure5_12'}
```

```

        % Load data

        filename = ['data',identifier];
        load(filename)

        % Select points in time, in [ s ], which are used for the
        % frequency analysis of the pressure oscillations

        tstart = 0.3;
        tend   = 1.3;
        N = 4096;
        t = linspace(tstart,tend,N+1);   % sample times, N intervals
        t(end) = [];            % do not include t = 1.3 -> N points

        freqRes = 1/(tend - tstart);  % frequency resolution
        freqFFT = (0:N-1)*freqRes;    % frequencies [ Hz ] from fft

        % Pre-allocate space

        ntauf = length(taufData);
        nTw   = length(TwData);
        nt    = length(t);

        Amp    = zeros(ntauf,nTw);
        Freq   = zeros(ntauf,nTw);
        YfMean = zeros(ntauf,nTw);

        FFT_P = zeros(nt,1);
        y     = zeros(nt,4);
        P     = zeros(nt,1);
        Yf    = zeros(nt,1);

        % The frequency information of the pressure oscillations
        % is contained in the entries 2:N/2 of the FFT.
        % To determine the pressure amplitudes and mean fuel mass
        % fraction, only the points in time > 1 sec. are used.

        Ifreq = 2:N/2;
        freqFFT = freqFFT(Ifreq);
        I = find((t > 1),1);  % index of first t > 1

        % Extract data for pressure from solution structure and
        % determine peak-to-peak amplitudes [ kPa ], mean fuel mass
        % fractions [ - ], and frequencies [ Hz ]

        for i = 1:ntauf
            for j = 1:nTw
                y(:,:)  = deval(t,soltaufTw{i,j}).';
                P(:,1)  = y(:,1);
                Yf(:,1) = y(:,4);
                Amp(i,j)    = max(P(I:end)) - min(P(I:end));
                FFT_P(:,1) = fft(P);
                [val,ind]  = max(abs(FFT_P(Ifreq)));
```

```
                                    % determine max. in periodogram
                Freq(i,j)   = freqFFT(ind);
                YfMean(i,j) = mean(Yf(I:end));
            end
        end
        Amp = Amp*s.P0*1e-3;  % pressure amplitude [ kPa ]

        % Save frequency and amplitude data

        filename = ['freqData',identifier];
        save(filename,'s','y0','taufData','TwData','Amp','Freq',...
            'YfMean')

    otherwise

        error('Unknown identifier choice.');

end  % switch
```

freqDataRichardsEa.m

C.4 Plotting graphs

The script file `plotRichardsEa.m` (see Listing C.4.1) contains the code that was used to plot the figures of Section 5.4, using the data generated with the scripts `dataRichardsEa` (see Section C.2) and `freqDataRichardsEa` (see Section C.3). By setting the MATLAB variable `identifier` in Line 23 to a certain string and running the script, data for the figure associated with that identifier is loaded, and a graph is plotted and saved under an appropriate name. The available choices for the identifier are listed in Table C.10. It mentions which figures are produced for those choices, and the filenames under which the figures are saved (i.e. the `.fig` files). The data files (extension: `.mat`) that are needed to plot the figures, are shown between angular brackets.

For Figures 5.6–5.8 amplitude information is needed, which was not generated with the scripts `dataRichardsEa` or `freqDataRichardsEa`. It is therefore computed in the script `plotRichardsEa` in this section. To be able to compare the amplitude information of Figure 5.6 directly with those of Figures 5.9–5.12, the amplitudes are determined in the exact same way, using the same sample times.

A last remark on the code of `plotRichardsEa.m`: labels for the contour plots of Figures 5.10–5.12 must be placed manually in the plots by pointing and clicking the mouse on contour lines. You continue by pressing the 'Return' key. The contour plots are always produced in pairs: part (a) of the figure (the amplitude information), and part (b) of the figure (the frequency information). It may happen that the second contour plot is hidden behind the first, when you are asked to place the labels manually. Just select the second figure window to put it in the foreground, and continue with placing labels.

Identifier	Filename ⟨needed⟩	Description
`'Figure5_3'`	`Figure5_3.fig` ⟨`dataFigures5_3-4-5.mat`⟩	Plots Figure 5.3.
`'Figure5_4'`	`Figure5_4.fig` ⟨`dataFigures5_3-4-5.mat`⟩	Plots Figure 5.4.
`'Figure5_5'`	`Figure5_5.fig` ⟨`dataFigures5_3-4-5.mat`⟩	Plots Figure 5.5.
`'Figure5_6'`	`Figure5_6.fig` ⟨`dataFigure5_6.mat`⟩	Plots Figure 5.6.
`'Figure5_7'`	`Figure5_7.fig` ⟨`dataFigures5_7-8.mat`⟩	Plots Figure 5.7.
`'Figure5_8'`	`Figure5_8.fig` ⟨`dataFigures5_7-8.mat`⟩	Plots Figure 5.8.
`'Figure5_9'`	`Figure5_9.fig` ⟨`freqDataFigure5_9.mat`⟩	Plots Figure 5.9.

`'Figure5_10'`	`Figure5_10a.fig`
	`Figure5_10b.fig`
	⟨`freqDataFigure5_10.mat`⟩
		Plots Figures 5.10a and 5.10b.

`'Figure5_11'`	`Figure5_11a.fig`
	`Figure5_11b.fig`
	⟨`freqDataFigure5_11.mat`⟩
		Plots Figures 5.11a and 5.11b.

`'Figure5_12'`	`Figure5_12a.fig`
	`Figure5_12b.fig`
	⟨`freqDataFigure5_12.mat`⟩
		Plots Figures 5.12a and 5.12b.

Table C.10: Identifier choices for the script file `plotRichardsEa.m` (see Line 23 of Listing C.4.1). The figures are saved as `.fig` files. Data that is needed to plot the figures, is loaded from the `.mat` files; they are indicated by angular brackets.

Listing C.4.1: Script file plotRichardsEa.m. Plots the figures from Section 5.4, using data generated with `dataRichardsEa.m` (see Section C.2) and `freqDataRichardsEa.m` (see Section C.3). The choice of the identifier set in Line 23 determines the graph that is plotted.

plotRichardsEa.m

```
 1  % plotRichardsEa.m
 2
 3  % Clear workspace, close figure windows, and clear command window
 4
 5  clear all
 6  close all
 7  clc
 8
 9  % Set identifier to plot a certain graph from the Interim Report,
10  % Chapter 5. Choose from:
11  %
12  %   'Figure5_3'  = plot Figure 5.3 from Interim Report
13  %   'Figure5_4'  = plot Figure 5.4 from Interim Report
14  %   'Figure5_5'  = plot Figure 5.5 from Interim Report
15  %   'Figure5_6'  = plot Figure 5.6 from Interim Report
16  %   'Figure5_7'  = plot Figure 5.7 from Interim Report
17  %   'Figure5_8'  = plot Figure 5.8 from Interim Report
18  %   'Figure5_9'  = plot Figure 5.9 from Interim Report
19  %   'Figure5_10' = plot Figure 5.10 from Interim Report
20  %   'Figure5_11' = plot Figure 5.11 from Interim Report
21  %   'Figure5_12' = plot Figure 5.12 from Interim Report
22
23  identifier = 'Figure5_3';
24
25  % Select data to load, plot requested figure, and save it
26
27  switch identifier
28
29      case 'Figure5_3'
30
31          % Load data
32
33          load dataFigures5_3-4-5
34
35          % Select time-span, time in [ s ]
36
37          tstart = 0;
38          tend   = 0.3;
39          t = linspace(tstart,tend,501);
40
41          % Plot graph
42
43          h = figure('Name','Figure 5.3 of Interim Report');
44          nTwData = length(TwData);
45
46          for i = 1:nTwData
47
48              % Extract data for pressure at specified points in time
```

```
            % from solution structure

            y = deval(t,solTw{i}).';
            P = y(:,1);    % pressure, [ - ]

            % Plot pressure versus time in subplot, and annotate

            subplot(nTwData,1,nTwData - i + 1)
            plot(t,P)      % time in [ s ], pressure in [ - ]
            axis([tstart tend 0.5 2])

            xlabel('Time (s)')
            ylabel({'Dimensionless','pressure'})
            text(0.25,1.75,0,...
                ['T_w = ',num2str(TwData(i))],'FontSize',9)

            % Plot horizontal, dashed line at P = 1 (for reference)

            hold on
            plot([tstart tend],[1 1],'k--')

        end

        % Save graph

        saveas(h,'Figure5_3','fig')

    case 'Figure5_4'

        % Load data

        load dataFigures5_3-4-5

        % Select time-span, time in [ s ]

        tstart = 0;
        tend   = 0.05;
        nt = 501;
        t  = linspace(tstart,tend,nt);

        % Extract data for temperature from solution structure;
        % collect data in array, with Tw varying over the columns

        TArray = zeros(nt,length(TwData));
        for i = 1:length(TwData)
            y = deval(t,solTw{i}).';
            TArray(:,i) = y(:,2);    % temperature, [ - ]
        end

        % Plot graph

        h = figure('Name','Figure 5.4 of Interim Report');
        plot(t,TArray)
        axis([tstart tend 1 15]);

        % Annotate graph

        for i = 1:length(TwData)
            strLegend{i} = ['T_w = ', num2str(TwData(i)), ' K'];
        end

        hleg = legend(strLegend{:});
        set(hleg,'FontSize',9)
        text(0.041,10.75,0,...
            ['\tau_f = ',num2str(round(1/s.itauf*1e3)),' ms'],...
            'FontSize',9)
        title('Figure 5.4')
        xlabel('Time (s)')
        ylabel('Dimensionless temperature')

        % Save graph

        saveas(h,'Figure5_4','fig')

    case 'Figure5_5'

        % Load data

        load dataFigures5_3-4-5

        % Select time-span, time in [ s ]

        tstart = 0;
        tend   = 0.05;
        nt = 501;
        t  = linspace(tstart,tend,nt);

        % Extract data for fuel mass fraction from solution
        % structure; collect data in array, with Tw varying over the
        % columns

        YfArray = zeros(nt,length(TwData));
        for i = 1:length(TwData)
            y = deval(t,solTw{i}).';
            YfArray(:,i) = y(:,4);    % fuel mass fraction, [ - ]
        end

        % Plot graph

        h = figure('Name','Figure 5.5 of Interim Report');
        plot(t,YfArray)
        axis([tstart tend 0 s.Yfi]);

        % Annotate graph

        for i = 1:length(TwData)
            strLegend{i} = ['T_w = ', num2str(TwData(i)), ' K'];
        end
        hleg = legend(strLegend{:},...
            'Location','NorthWest');
        set(hleg,'FontSize',9)
        text(0.007,0.042,0,...
            ['\tau_f = ',num2str(round(1/s.itauf*1e3)),' ms'],...
            'FontSize',9)
        title('Figure 5.5')
        xlabel('Time (s)')
        ylabel('Fuel mass fraction')

        % Save graph

        saveas(h,'Figure5_5','fig')

    case 'Figure5_6'

        % Load data

        load dataFigure5_6

        % Select points in time, in [ s ], which are used to
        % determine the peak-to-peak pressure amplitude (i.e, use
        % the points in time consistent with those used for Figures
        % 5.9-5.12 of the Interim Report)

        tstart = 0.3;
        tend   = 1.3;
        N = 4096;
        t = linspace(tstart,tend,N + 1);  % N intervals
        I = find((t > 1),1);              % index first t > 1
        t = t(I:(end - 1));               % time span from 1 to 1.3,
                                          % not including t = 1.3

        % Extract data for pressure from solution structure,
        % and determine peak-to-peak amplitude, in [ kPa ]

        nh  = length(hData);
        Amp = zeros(nh,1);

        for i = 1:nh
            y = deval(t,solh{i}).';
            Amp(i) = max(y(:,1)) - min(y(:,1));  % amplitude, [ - ]
        end
        Amp = s.P0*Amp*1e-3;  % peak-to-peak amplitude [ kPa ]

        % Plot graph

        h = figure('Name','Figure 5.6 of Interim Report');
        plot(hData,Amp)

        axis([100 165 0 80]);
        set(gca,'Box','on','TickDir','out')

        % Annoatate, indicate areas of steady combustion and flame
        % extinction

        hold on
        fill([100 110 110 100],[0 0 80 80],[.6 .6 .6])
        text(105,40,'Steady combustion',...
            'Rotation',90,...
            'HorizontalAlignment','center')
        fill([155 165 165 155],[0 0 80 80],[.6 .6 .6])
        text(160,40,'Flame extinction',...
            'Rotation',90,...
            'HorizontalAlignment','center')

        title('Figure 5.6')
        xlabel('Heat transfer coefficient (W/m^2/K))')
        ylabel('Amplitude (peak-to-peak, in kPa)')

        % Save graph

        saveas(h,'Figure5_6','fig')

    case 'Figure5_7'

        % Load data

        load dataFigures5_7-8

        % Select time-span, time in [ s ]

        tstart = 0.9;
        tend   = 0.92;
        t = linspace(tstart,tend,501);
        tplot  = (t - tstart)*1e3;    % time in [ ms ]

        % Plot graph

        h = figure('Name','Figure 5.7 of Interim Report');
        nfData = length(fData);

        for i = 1:nfData

            % Extract data for pressure and velocity at specified
            % points in time from solution structure

            y = deval(t,solf{i}).';
            P = y(:,1);    % pressure, [ - ]
            u = y(:,3);    % velocity, [ - ]

            % Compute pressure amplitude, normalized pressure,
            % and normalized velocity

            Amp = s.P0*(max(P) - min(P))*1e-3;  % pressure amplitude
                                                % in [ kPa ]
            P = P/max(P);  % pressure normalized to its maximum
            u = u/max(u);  % velocity normalized to its maximum

            % Plot pressure and velocity versus time in subplot,
            % and annotate

            subplot(length(fData),1,nfData - i + 1)
            plot(tplot,[P u])   % pressure and velocity,
                                % time in [ ms ]

            axis([tplot(1) tplot(end) -0.5 1.1])
            hleg = legend('Normalized pressure',...
                'Normalized velocity','Location','NorthOutside');
            set(hleg,'FontSize',9)
            xlabel('Time (ms)')
            ylabel({'Normalized pressure','and velocity'})
            text(0.5,-0.3,0,...
                {['f = ',num2str(fData(i))],...
                ['Pressure amplitude = ',num2str(round(Amp)),...
                ' kPa']},...
                'FontSize',9)

            % Plot horizontal, dashed line at y = 0 (for reference)

            hold on
            plot([tplot(1) tplot(end)],[0 0],'k--')

        end % for

        % Save graph

        saveas(h,'Figure5_7','fig')

    case 'Figure5_8'

        % Load data

        load dataFigures5_7-8

        % Select time-span, time in [ s ]

        tstart = 0.9;
        tend   = 0.92;
        t = linspace(tstart,tend,501);
        tplot  = (t - tstart)*1e3;    % time in [ ms ]

        % Plot graph

        h = figure('Name','Figure 5.8 of Interim Report');
        nfData = length(fData);

        for i = 1:nfData

            % Extract data for pressure and fuel mass fraction at
            % specified points in time from solution structure

            y  = deval(t,solf{i}).';
            P  = y(:,1);    % pressure, [ - ]
            Yf = y(:,4);    % fuel mass fraction, [ - ]

            % Compute pressure amplitude, normalized pressure,
            % and normalized fuel mass fraction

            Amp = s.P0*(max(P) - min(P))*1e-3;  % pressure amplitude
                                                % in [ kPa ]
            P  = P/max(P);   % pressure normalized to its maximum
            Yf = Yf/max(Yf); % fuel mass fraction normalized to its
                             % maximum

            % Plot pressure and fuel mass fraction versus time in
            % subplot, and annotate

            subplot(length(fData),1,nfData - i + 1)
            plot(tplot,[P Yf]) % pressure and fuel mass fraction,
                               % time in [ ms ]

            axis([tplot(1) tplot(end) 0 1.1])
            hleg = legend('Normalized pressure',...
                'Normalized fuel mass fraction',...
                'Location','NorthOutside');
            set(hleg,'FontSize',9)
            xlabel('Time (ms)')
            ylabel({'Normalized pressure','and fuel mass fraction'})
            text(0.5,0.2,0,...
                {['f = ',num2str(fData(i))],...
                ['Pressure amplitude = ',num2str(round(Amp)),...
                ' kPa']},...
                'FontSize',9)

        end % for

        % Save graph

        saveas(h,'Figure5_8','fig')

    case 'Figure5_9'

        % Load data

        load freqDataFigure5_9

        % Plot graph

        h = figure('Name','Figure 5.9 of Interim Report');
        plot(LtpData,[Freq,Amp])
        legend('Frequency','Amplitude','Location','North')
        axis([0.475 0.825 0 250]);
        set(gca,'Box','on','TickDir','out')

        % Annoatate, indicate areas of steady combustion and flame
        % extinction

        hold on
        fill([0.475 0.525 0.525 0.475],[0 0 250 250],[.6 .6 .6])
        text(0.5,125,'Steady combustion',...
            'Rotation',90,...
            'HorizontalAlignment','center')
        fill([0.775 0.825 0.825 0.775],[0 0 250 250],[.6 .6 .6])
        text(0.8,125,'Flame extinction',...
            'Rotation',90,...
            'HorizontalAlignment','center')

        title('Figure 5.9')
        xlabel('Tailpipe length (m)')
        ylabel('Amplitude (kPa) and frequency (Hz)')

        % Save graph

        saveas(h,'Figure5_9','fig')

    case {'Figure5_10','Figure5_11','Figure5_12'}

        % Load data

        filename = ['freqData',identifier];
        load(filename)

        % Plot graph, part (a) contour plot of amplitudes

        strFigure = ['Figure 5.',identifier(end-1:end),'(a)'];
        h = figure('Name',[strFigure,'a of Interim Report']);

        % Plot pressure amplitude contour lines

        [XTw,Ytauf] = meshgrid(TwData,taufData);
        [C,hC] = contour(XTw,Ytauf,Amp,[0:10:100],'k');
        axis([700 1400 0.01 0.09])

        % Add lines indicating boundary between steady and
        % oscillatory combustion

        hold on
        contour(XTw,Ytauf,Amp,0.01*s.P0*1e-3*[1 1],'k',...
            'LineWidth',2)

        % Add lines indicating boundary between combustion and
        % flame extinction

        contour(XTw,Ytauf,YfMean,0.99*s.Yfi*[1 1],'k--',...
            'LineWidth',2)

        % Annotate graph

        title(strFigure)
        xlabel('Wall temperature (K)')
        ylabel('Flow time (s)')
        text(1200,0.05,'Steady combustion')
        text(750,0.015,'Flame extinction')

        % Label contour lines manually by pointing and clicking with
        % the mouse (press <Return> to continue)

        clabel(C,hC,'manual','FontSize',7)

        % Save graph, part (a) contour plot of amplitudes

        saveas(h,[identifier,'a'],'fig')
        %close(h)

        % Plot graph, part (b) contour plot of frequencies

        strFigure = ['Figure 5.',identifier(end-1:end),'(b)'];
        h = figure('Name',[strFigure,'b of Interim Report']);

        % Plot pressure frequency contour lines

        [XTw,Ytauf] = meshgrid(TwData,taufData);
        [C,hC] = contour(XTw,Ytauf,Freq,[70:10:200],'k');
        axis([700 1400 0.01 0.09])

        % Add lines indicating boundary between steady and
        % oscillatory combustion

        hold on
        contour(XTw,Ytauf,Amp,0.01*s.P0*1e-3*[1 1],'k',...
            'LineWidth',2)

        % Add lines indicating boundary between combustion and
        % flame extinction

        contour(XTw,Ytauf,YfMean,0.99*s.Yfi*[1 1],'k--',...
            'LineWidth',2)

        % Annotate graph

        title(strFigure)
        xlabel('Wall temperature (K)')
        ylabel('Flow time (s)')
        text(1200,0.05,'Steady combustion')
        text(750,0.015,'Flame extinction')

        % Label contour lines manually by pointing and clicking with
        % the mouse (press <Return> to continue)

        clabel(C,hC,'manual','FontSize',7)

        % Save graph

        saveas(h,[identifier,'b'],'fig')

    otherwise

        error('Unknown identifier choice.');

end  % switch
```

plotRichardsEa.m

Appendix D

MATLAB codes of auxiliary functions

D.1 Adjusting model parameter structure

The function `setparam` (see Listing D.1.1) was written to specify the contents of multiple fields of a structure in an easy way. It also checks if the fieldnames exist for which the contents is to be set, and it returns an error message if that is not the case. So, by using the function `setparam`, the contents of multiple fields of a structure can be set in one go, and the user is sure that the correct fields are modified. However, since the input arguments of the function are not checked for correctness, the user should take care that the input is specified in the intended way. Otherwise, an unexpected error message may appear.

The input of the function `setparam` should be of the following form: a structure (say, `s`), followed by a paired list of strings of fieldnames and values (say, `...,'fld`*n*`',val`*n*`,...`). Thus, the input is: `s,'fld1',val1,...`. The function then returns an identical structure, except that the fields with the names `fld`*n* are given the values `val`*n*.

The function `setparam` is used by the functions `setKilicarslan` (see Section A.1), `setAhrensEa` (see Section B.1), and `setRichardsEa` (see Section C.1).

For completeness, an overview of the MATLAB functions defined in this section, is presented in Table D.1.

Function	Input	Description
`setparam`	`s,'fld1',val1,...`	Returns a structure equal to `s`, except that the fields `fld`*n* are given the specified values `val`*n*.

Table D.1: Overview of the functions that are defined in Section D.1.

Listing D.1.1: Function setparam. Given a structure and a paired list of field-names and values, it returns an identical structure, except that the specified fields are set to the corresponding values.

setparam.m

```
function sOut = setparam(sIn,varargin);

% Copy input structure

sOut = sIn;

% Adjust user specified fields to user specified values

for i = 1:2:length(varargin)
    if ~isfield(sOut,varargin{i})
        error('Unknown parameter.');
    end
    sOut = setfield(sOut,varargin{i},varargin{i+1});
end
```

setparam.m

This report was provided with a CD-ROM in the back, on which all the MATLAB codes can be found that were presented in the appendices.