Warsaw University of Technology

FACULTY OF
POWER AND AERONAUTICAL ENGINEERING

Institute of Heat Engineering

# MKWS Project

Products and charactaristics of combustion of self igniting mixture under constant volume

Michał Matczak
student record book number 304290

project supervisor
Mateusz Żbikowski

WARSAW 2022

# Contents

# 1. Introduction

The aim of this project is to analyze products of combustion in enclosed container, given initial pressure, temperature and well mixed mixture of fuel and oxidizer. The variables that will differentiate one case from another are the mass proportions and types of fuel and oxidizer that are being used. Solving such problem might lead to very interesting results. It can be also ground to base future developments of e.g. pressurized tank or even engines. Depending on the completeness of combustion the remaining gasous and sometimes solid material may lead to deterioration of real vessels. It can also have an application while discussing pollution and emmision of gases.

# 2. Model

The model is very simple to comprehend. We have a tank of unknown volume - which helps us make the problem more broad. Model assumes self ignition of the flammable medium, thats why the temperature at the beginning has to be relatively high. Only thing under our control are initial pressure, initial temperature and mass fraction of fuel - type of setting one can find in combustion laboratories. Mathematical model gives us freedom to not worry about costs of acquiring gases for the test.
During mine calculation the variables were as such:

- $p_0$ = 1 atm
- $T_0$= 1000 K

## 2.1. Code

The code allows us to choose from 6 diffrent types of fuels and 2 types of oxidizer. The user can also control initial parameters, as it was said in **2. Model**. Number of iterations - both for the time that elapses after ignition and the amount of points between two extreme scenarios (only oxidizer and only fuel) can be adjusted. The end result are 3 figures - max pressure, temperature and products, dependent on the composition of a mixture.

# 3. Results

## 3.1. Hydrogen and Oxygen

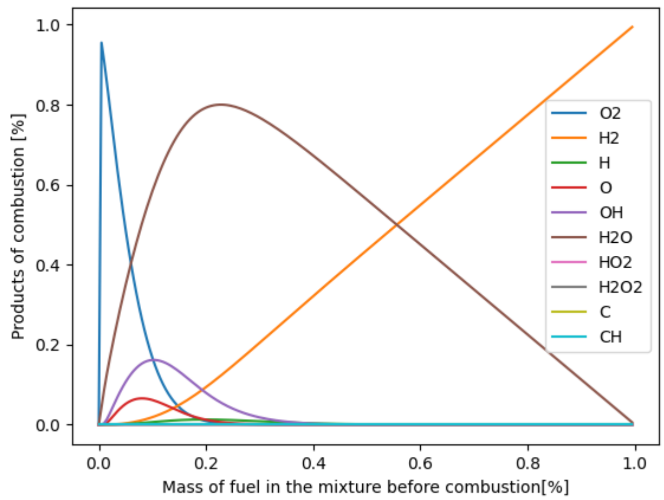


**Figure 3.1.** 10 products with highest mass participation

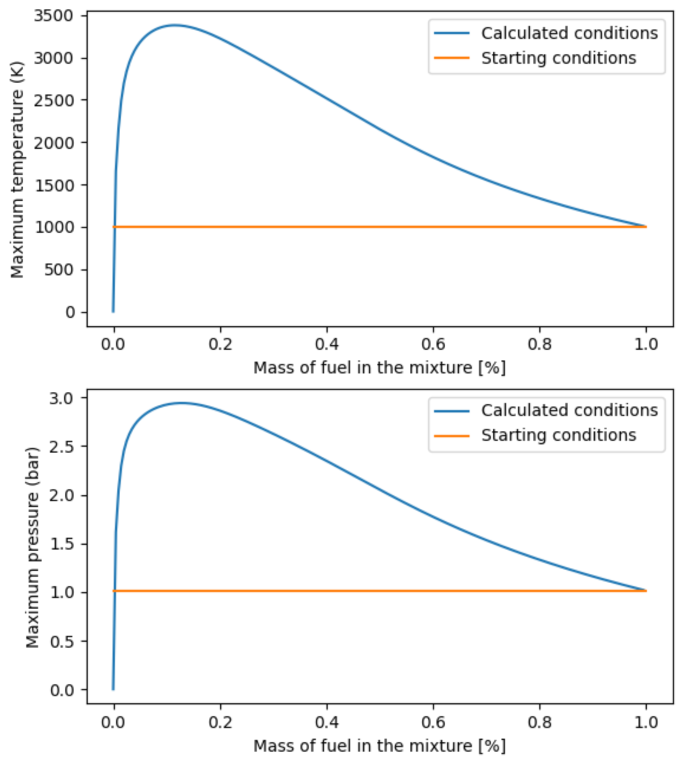


**Figure 3.2.** Range of max temperature and pressure dependent on proportions

## 3.2. Methane and Oxygen

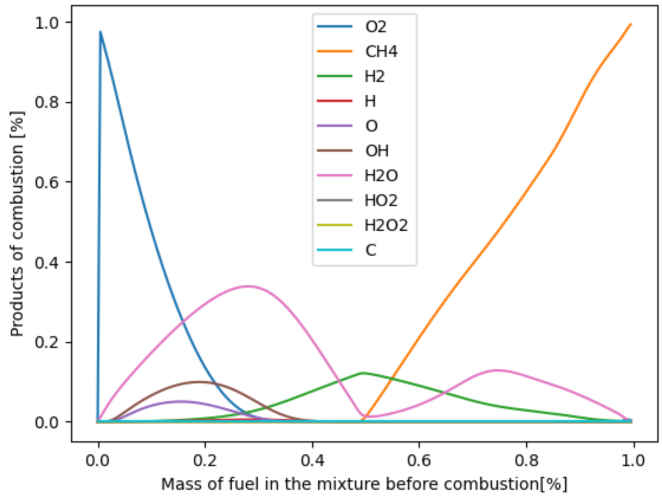


**Figure 3.3.** 10 products with highest mass participation

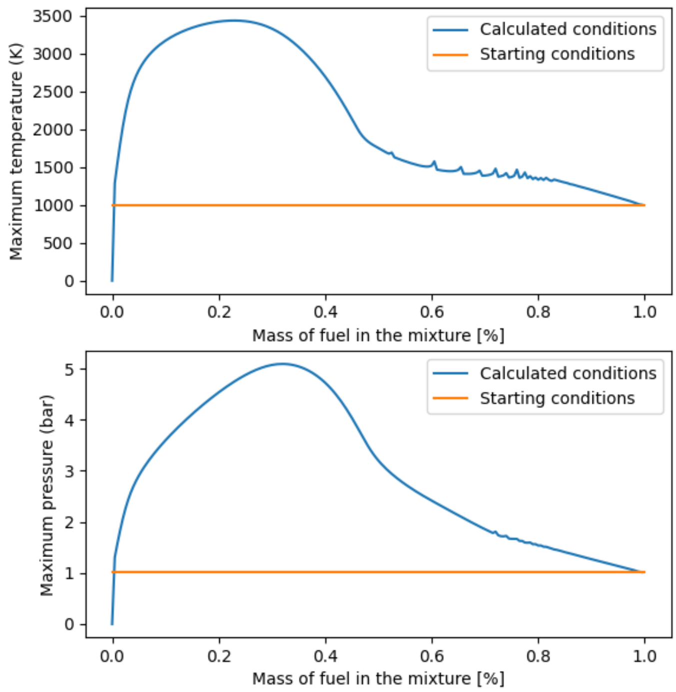


**Figure 3.4.** Range of max temperature and pressure dependent on proportions

## 3.3. Ethane and Oxygen

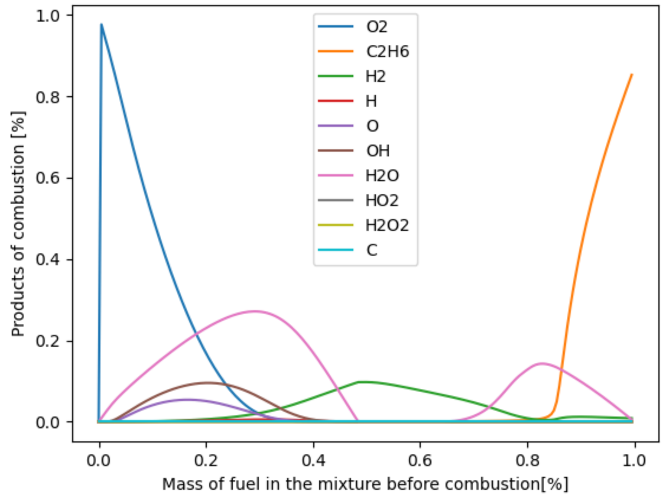


**Figure 3.5.** 10 products with highest mass participation

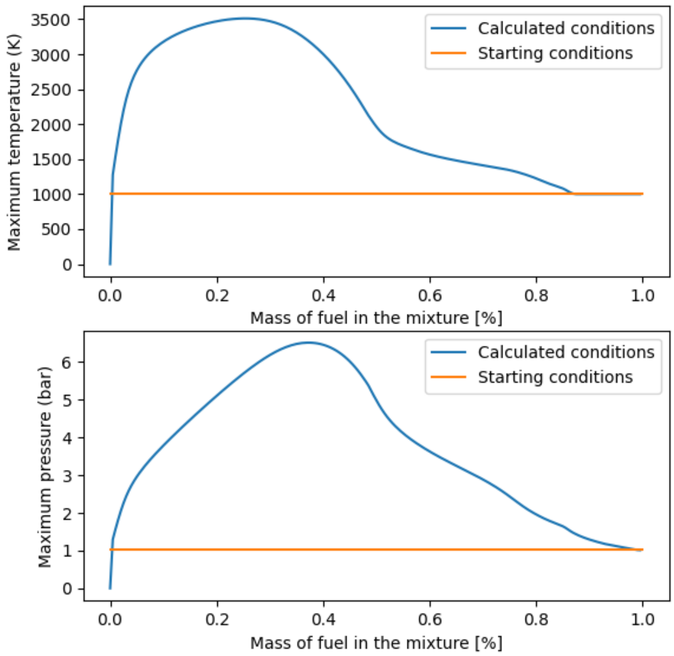


**Figure 3.6.** Range of max temperature and pressure dependent on proportions


6

### 3.4. Carbon monoxide and Air

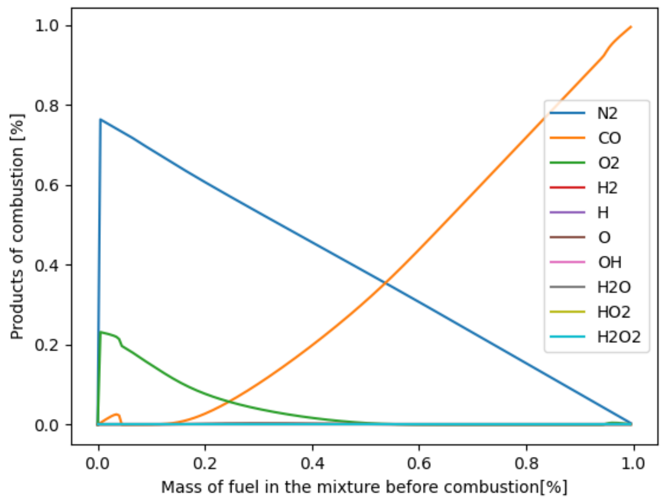


**Figure 3.7.** 10 products with highest mass participation

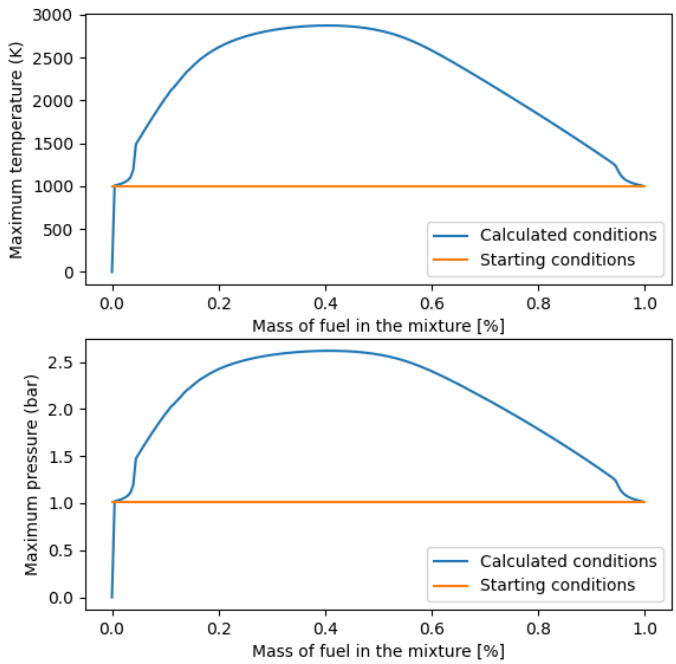


**Figure 3.8.** Range of max temperature and pressure dependent on proportions



7

## 3.5. Propane and Air

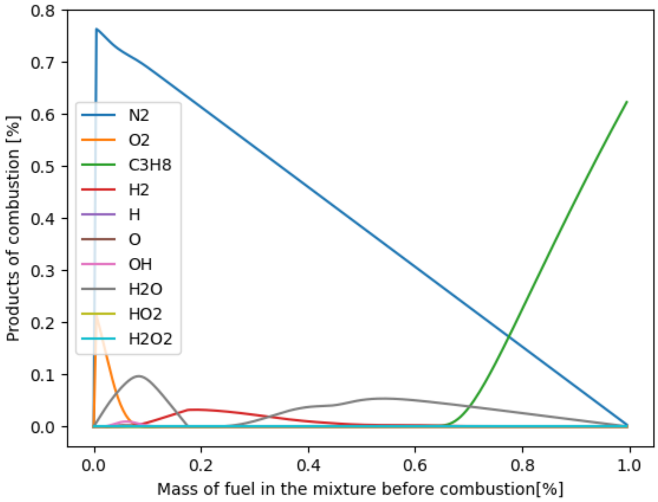


**Figure 3.9.** 10 products with highest mass participation

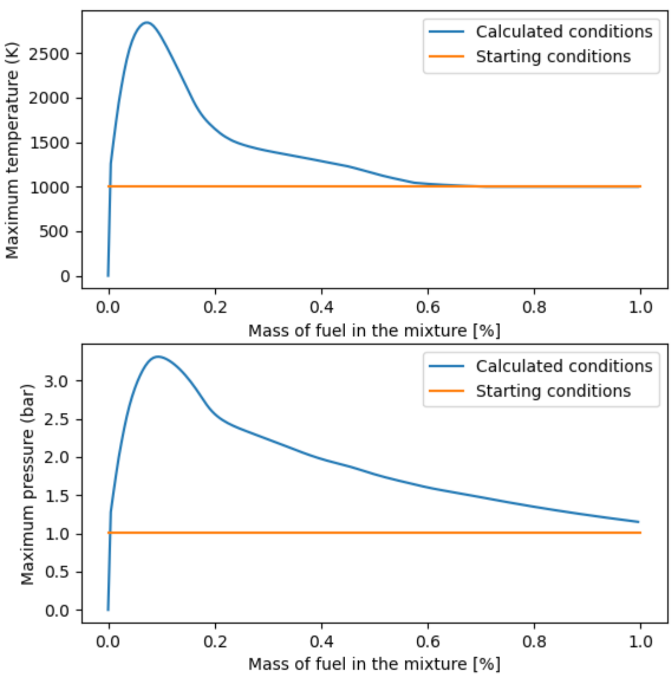


**Figure 3.10.** Range of max temperature and pressure dependent on proportions

## 3.6. Methanol and Air

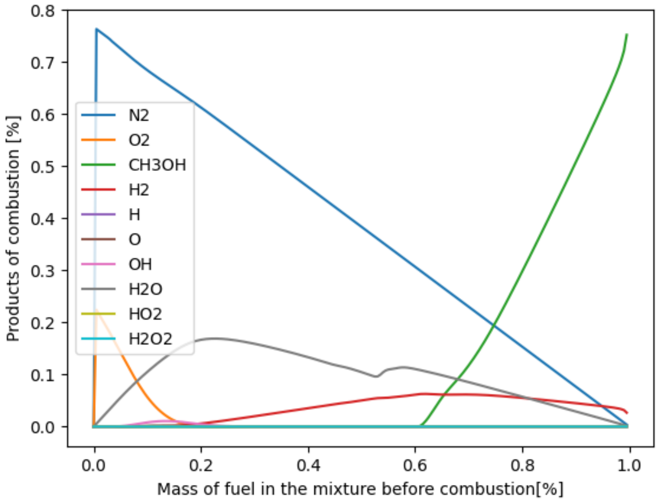


**Figure 3.11.** 10 products with highest mass participation

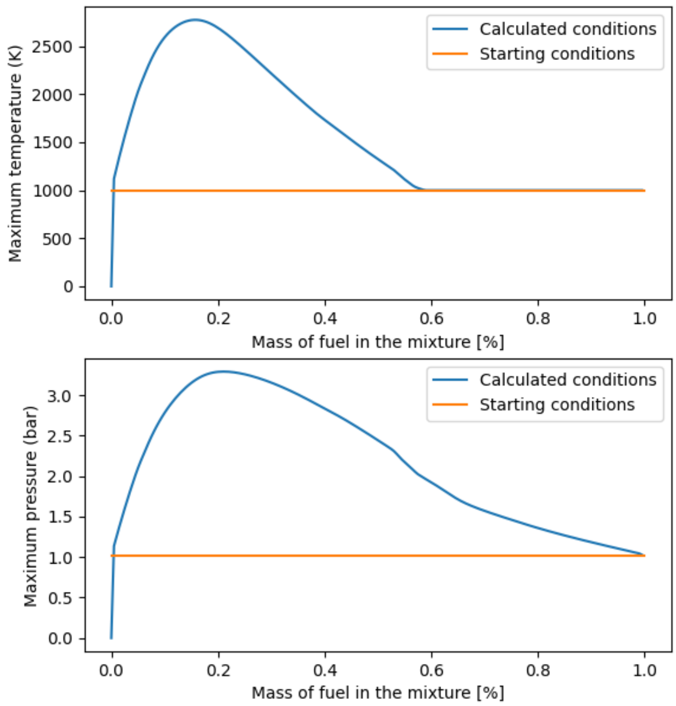


**Figure 3.12.** Range of max temperature and pressure dependent on proportions

# 4. Conclusion

After analyzing given diagrams we can see that:

- more chemicaly complex mixtures tend to have more side products of combustion
- in nearly every case there is a space on diagram where original substrates are not present in the combustion aftermath
- simpler mixtures tend to have peaks of pressure and temperature closer to the middle of diagrams (where there is a similar amount of oxidizer and fuel)
- more complex mixtures have their peaks closer to the point when oxidizer is a majority
- combustion of carbon monoxide isn't very efficient
- nitrogen that is in the air rarely reacts during the combustion process
- very often byproduct of combustion is water (in it's gasous form)


10

# References


[1] https://cantera.org/documentation/dev/sphinx/html/cython/thermo.html

[2] https://www.geeksforgeeks.org/python-program-for-bubble-sort/

[3] http://fluid.wme.pwr.wroc.pl/~spalanie/dydaktyka/combustion_en/tut_combustion_en.pdf

[4] ***CANTERA Tutorials*** *A series of tutorials to get started with the python interface of Cantera version 2.1.1 by Anne Felden*

[5] https://www.sciencedirect.com/topics/engineering/specific-gas-constant

# List of Symbols and Abbreviations

# List of Figures



# List of Tables

# List of Appendices

# Appendix 1. Code

```
import numpy as np
import cantera as ct
import matplotlib.pyplot as plt
import csv
from alive_progress import alive_bar
########################################################
#defining what air is
air = "O2:0.21,N2:0.79"
#simulation options
Duration = 250
iter1=200 #Quality of products of combustion calculations
iter2=500 #Quality of pressure and temperature calculations
plot_gen = True
Outside_data = ''
#Idle variables (placeholders)
low_limit = 0
high_limit = 1
low_limit_fig=0
high_limit_fig=1
ox_R = 280
fuel_R = 280

#Mechanism used for the process
gas = ct.Solution('gri30.yaml')
#Gas state
P_0 = ct.one_atm
T_0 = 1000
gas.TP = T_0, P_0
#Choosing oxidizer and fuel
fuel = input('Fuel:\n air and O2 - H2, CO, CH4, C2H6, C3H8\n ...
...only air - CH3OH\n Choose: ')
if fuel == 'CH3OH':
    oxidizer = 'air'
else:
    oxidizer = input('Oxidizer (air or O2): ')

#Loading flammability limits
if oxidizer == 'O2':
    Outside_data = 'O2.csv'
```

```
    ox_R = 260
elif oxidizer == 'air':
    Outside_data = 'AIR.csv'
    oxidizer = air
    ox_R = 287
with open(Outside_data) as f:
    reader = csv.reader(f, delimiter='\t')
    Limits = [(col1, float(col2), float(col3),float(col4))
                for col1, col2, col3, col4 in reader]
for a in range(len(Limits)):
        arr_limits=np.asarray(Limits)
        if arr_limits[a,0] == fuel :
            low_limit= float(arr_limits[a,1])
            high_limit = float(arr_limits[a, 2])
            fuel_R = float(arr_limits[a, 3])

#converting volume fractions to mass fractions
low_limit_fig= (low_limit/fuel_R) /(((100-low_limit)/ox_R)+...
...+(low_limit/fuel_R))
high_limit_fig= (high_limit/fuel_R) /(((100-high_limit)/ox_R)+...
...+(high_limit/fuel_R))
low_limit = 0
high_limit =1

#Allocating space for data
fo = 0 #Fuel to all Mixture ratio
fos=np.zeros(iter1)
data = np.zeros((iter1,10))
data1 = np.zeros((iter1,2))
arr3 = np.zeros(10)
#Main working loop
with alive_bar(iter1,force_tty=True) as bar:
    for m in range(iter1):
        fo += (high_limit-low_limit)/iter1
        if fo >=high_limit:
            fo=high_limit-0.0001
        #Main calculations
        gas.TP = T_0, P_0
        gas.set_mixture_fraction(fo, fuel, oxidizer)
        r = ct.IdealGasReactor(gas)
        sim = ct.ReactorNet([r])
        time = Duration/iter2
```

```
        times = np.zeros(iter2)

        #Getting max T and max P
        p_max = P_0
        T_max = T_0
        for n in range(iter2):
            time += Duration/iter2
            sim.advance(time)
            times[n] = time # time in s
            if T_max < r.T:
                T_max = r.T
            if p_max < r.thermo.P:
                p_max = r.thermo.P
# Bubble Sort for 10 most present substances
        if m ==0:
            gas.set_equivalence_ratio(1.0, fuel, oxidizer)
            name = gas.species_names
            string2 = gas.Y
            arr1 = np.array(name)
            arr2 = np.array(string2)
            n = len(arr2)
            swapped = False
            for i in range(n - 1):
                for j in range(0, n - i - 1):
                    if arr2[j] < arr2[j + 1]:
                        swapped = True
                        arr2[j], arr2[j + 1] = arr2[j + 1], arr2[j]
                        arr1[j], arr1[j + 1] = arr1[j + 1], arr1[j]
            fo=low_limit
            fos[m] = fo
            arr3=arr1[:10]
        else:
            fos[m] = fo
            data[m,0:] = r.thermo[arr3].Y
            data1[m,0] = T_max
            data1[m,1] = p_max
        bar()

# Plot the results if matplotlib is installed.
if plot_gen==True:
    plt.xlabel('Mass of fuel in the mixture before combustion[%]')
    plt.ylabel('Products of combustion [%]')
```

```
for o in range(10):
    plt.plot(fos, data[:,o], label=arr3[o])
plt.legend()
plt.show()

plt.clf()
plt.subplot(2,1, 1)
plt.plot(fos, data1[:, 0], label='Calculated␣conditions')
plt.plot([0,1], [T_0,T_0], label='Starting␣conditions')
#plt.plot([low_limit_fig, low_limit_fig], [T_0,T_max], color='C3')
#plt.plot([high_limit_fig, high_limit_fig], [T_0,T_max], color='C3')
plt.xlabel('Mass␣of␣fuel␣in␣the␣mixture␣[%]')
plt.ylabel('Maximum␣temperature␣(K)')
plt.legend()
plt.subplot(2,1, 2)
plt.plot(fos, data1[:, 1]/100000, label='Calculated␣conditions')
plt.plot([0,1], [P_0/100000,P_0/100000],label='Starting␣conditions')
#plt.plot([low_limit_fig, low_limit_fig], [P_0/100000,p_max/100000],...
...color='C3')
#plt.plot([high_limit_fig, high_limit_fig], [P_0/100000,p_max/100000],...
...color='C3')
plt.xlabel('Mass␣of␣fuel␣in␣the␣mixture␣[%]')
plt.ylabel('Maximum␣pressure␣(bar)')
plt.legend()
plt.show()
```