presentation

October 23, 2023

1 Termopy: Modeling Thermodynamic Processes and Cycles

1.1 Introduction

Welcome to Termopy, a Python library designed to assist in the calculation and visualization of thermodynamic processes and cycles. Whether you're studying physics (FYS102) or working on projects related to thermodynamics, Termopy is a powerful tool that can help you analyze and represent the behavior of ideal gases during various processes.

1.2 Key Features

Termopy is organized into two main modules: the process package and the main package. These modules work together to provide a range of functionality, allowing you to examine different states of an ideal gas as it undergoes various processes. Termopy is designed to be user-friendly, and it automatically checks for numerical errors to ensure the accuracy of your results.

1.3 How It Works

1.3.1 Object-Oriented Approach

Termopy takes an object-oriented approach to thermodynamic modeling. At the heart of the library is the concept of a "fluid," which represents the working gas. To start using Termopy, you'll create a fluid object and provide it with essential information to determine the initial state of the gas. You can specify the gas type by name, choose from a list of 80 pre-defined gases stored in the library, or simply label it as "monatomic" or "diatomic," which will default to Helium and Nitrogen-2, respectively. If no gas type is specified, Termopy assumes the working gas is air.

1.3.2 Simulating Processes

Once you've defined your fluid object, you can simulate a variety of processes. Termopy offers predefined methods for isothermal, isobaric, isochoric, and adiabatic processes, with the potential for additional process types to be added in the future. To simulate a process, you'll need to specify an endpoint or a final state for a gas property such as pressure, volume, or temperature. Keep in mind that certain processes, like isothermal, cannot be defined by their final temperature due to their static nature.

1.3.3 State Vector

Every time you invoke one of the process methods, Termopy generates a state vector within the fluid object. This state vector records changes in internal variables over time, allowing you to

closely examine the behavior of the gas during the simulated process.

1.3.4 Visualization

To make your analysis even more insightful, Termopy provides built-in methods for visualization. You can use the "plot()" and "show()" functions, which extend the capabilities of the popular Matplotlib library. These functions simplify the programming process, making it easy for users to create plots and save them for further reference.

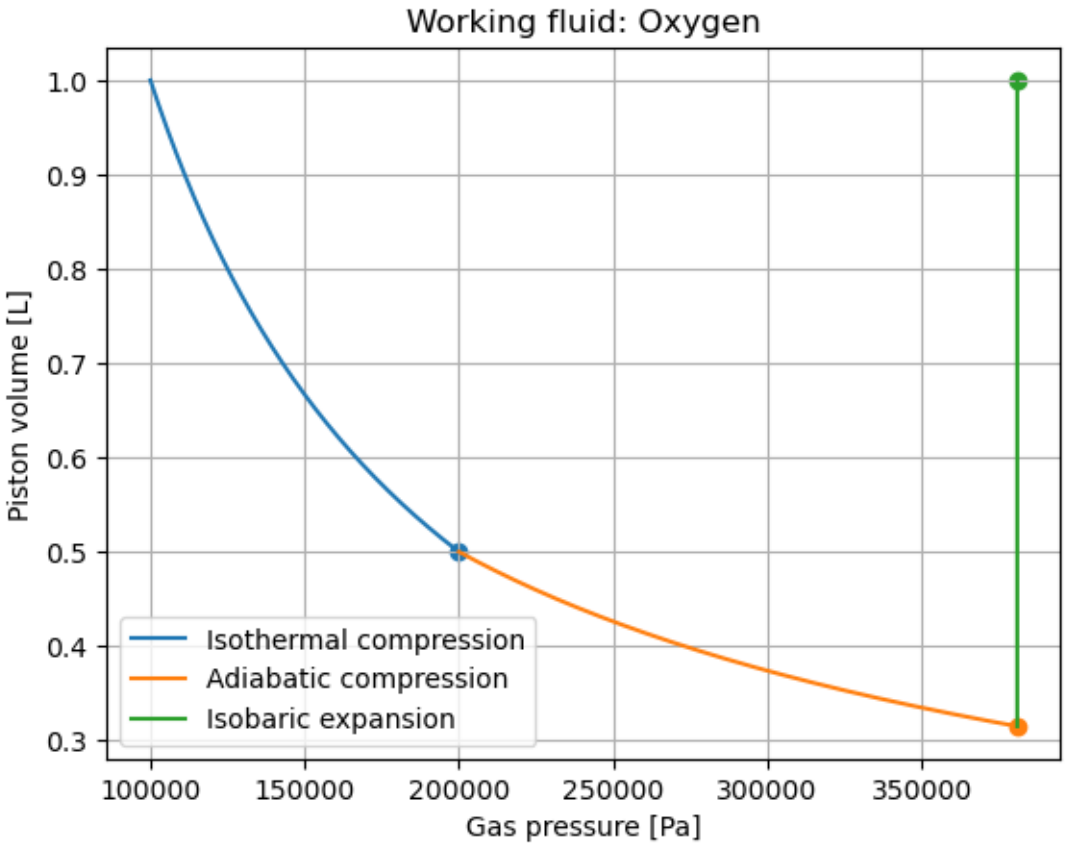
In conclusion, Termopy is a versatile and user-friendly tool that simplifies the modeling of thermodynamic processes and cycles. With its object-oriented design and visualization capabilities, it empowers you to gain a deeper understanding of ideal gases and their behavior under various conditions.

[2]:
```
import termoPy as tp

O2 = tp.Fluid(P=1e5, V=1e-3, T=500, gas='O2')

O2.isothermal(P=2e5)
O2.adiabatic(T=600)
O2.isobaric(V=1e-3)
O2.isochoric

tp.plot(O2, display="PV")
tp.show()
```



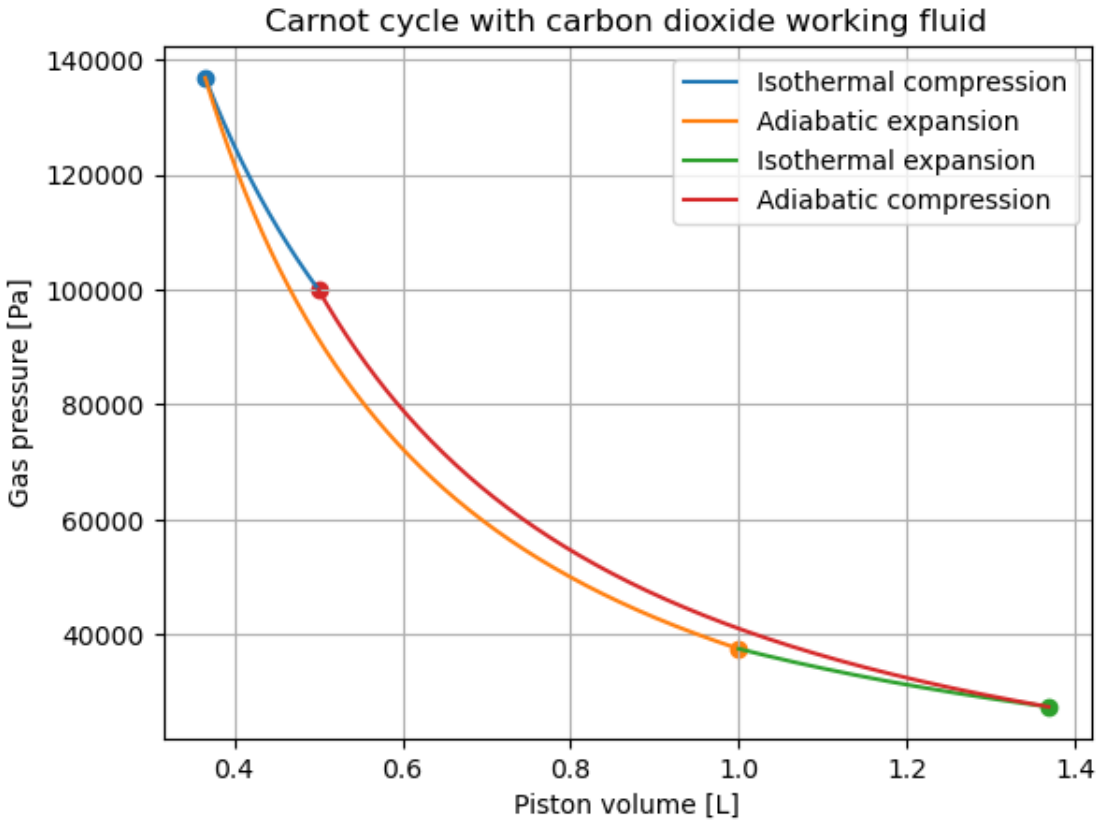
1.4 Modeling Thermodynamic Cycles

Termopy goes beyond individual processes and provides the ability to model complete thermodynamic cycles. It includes the following cycles: Carnot cycle, Stirling cycle, and Otto cycle. These cycles are implemented as classes, and they extend the capabilities of the library.

1.4.1 Carnot Cycle

The Carnot cycle is a fundamental thermodynamic cycle that represents the theoretical maximum efficiency for a heat engine operating between two temperature reservoirs. To create a Carnot cycle using Termopy, you can use the `Carnot` class. Here's an example of how to use it:

[3]:
```
carnot = tp.
  ↪Carnot(T_hot=400,T_cold=300,compression_ratio=2,P=1e5,V=1e-3,gas='CO2')
tp.plot(carnot,display="VP")
tp.show()
```



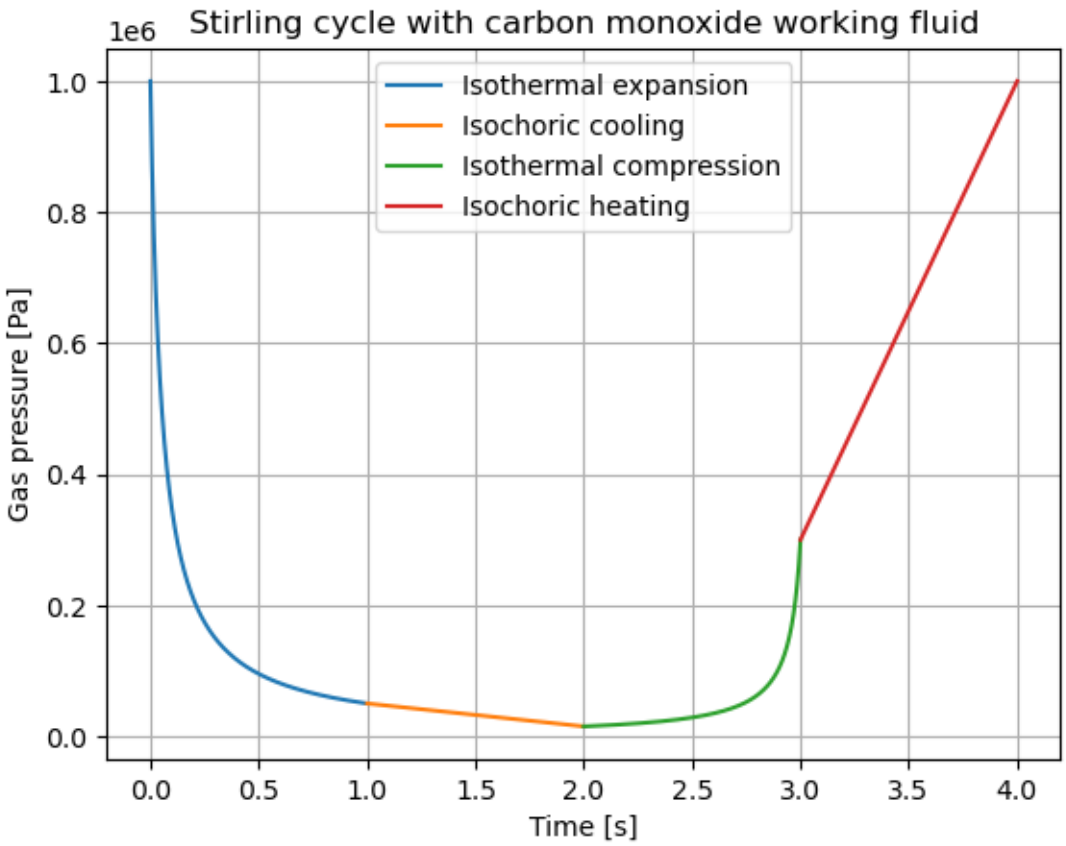
[4]:
```
stirling = tp.
  ↪Stirling(T_hot=1000,T_cold=300,compression_ratio=20,P=1e6,V=1e-3,gas='CO')

print(tp.version)

print(stirling.efficiency, stirling.COP)

tp.plot(stirling,display="P")
tp.show()
```

```
1.1.2
0.4418643222699116 2.2631381390171454
```



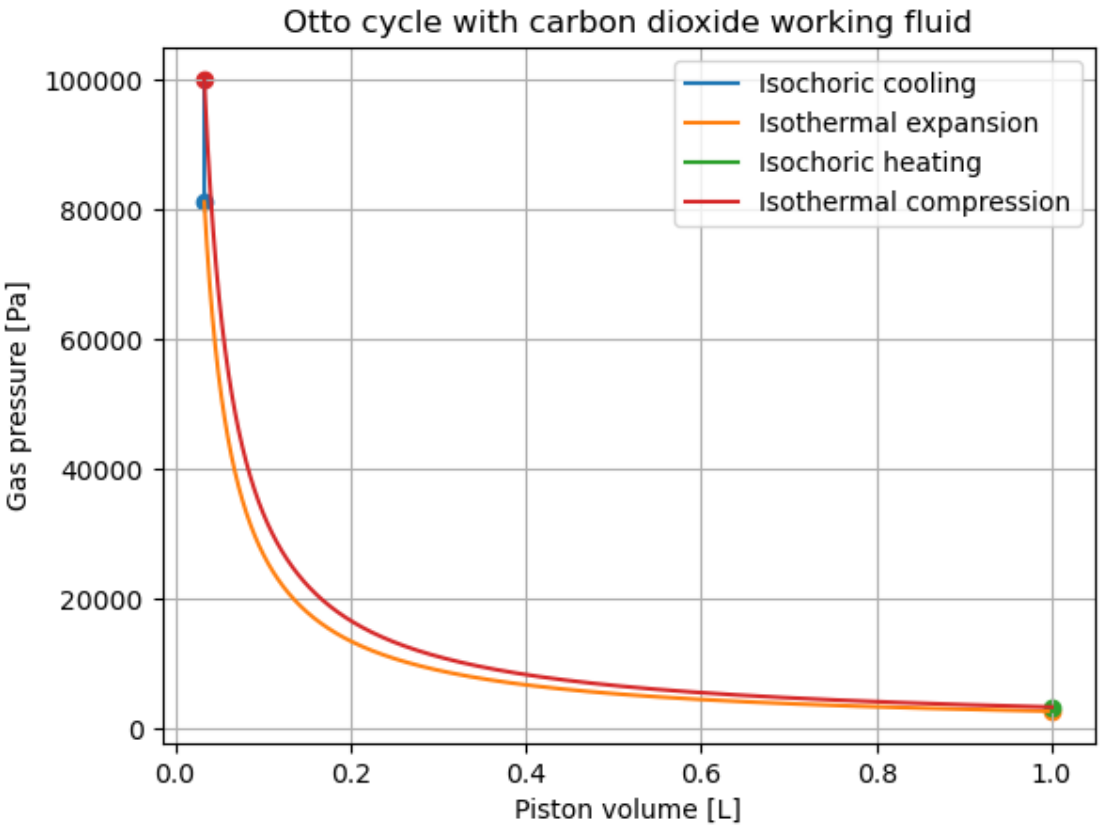
[5]:
```
otto = tp.Otto(T_hot=370,T_cold=300,compression_ratio=30,P=1e5,V=1e-3,gas='CO2')

tp.plot(otto,display="VP")

print(otto.efficiency)

tp.show()
```

```
0.18815313310664308
```



2 Termopy: Fluid Object Variables and Functions

The Termopy library provides a Fluid object that allows you to model and analyze thermodynamic processes. This documentation outlines the variables and functions available within the Fluid object for your reference.

2.1 Fluid Object Variables

2.1.1 Work (Variable: Scalar)

The `Work` variable represents the work done on the gas during the entire process. It is calculated through numerical integration of the pressure-volume (PV) curve. This variable provides insights into the energy exchange during the process.

2.1.2 Heat (Variable: Vector)

The `Heat` variable is a vector that represents the heat absorbed by the gas throughout the process. It stores the cumulative heat absorbed by the gas at each point in time. Analyzing this variable helps in understanding the heat transfer within the system.

2.1.3 Entropy (Variable: Vector)

The `Entropy` variable is a vector that tracks the changes in entropy over time during the process. Entropy is a fundamental thermodynamic property that can be used to assess the irreversibility of a process.

2.1.4 Time Taken (Variable: Scalar)

The `Time Taken` variable provides the total duration of the process in seconds. It is the sum of the time taken for each sub-process within the overall thermodynamic process.

2.1.5 Heat Added (Variable: Scalar)

The `Heat Added` variable is a scalar that represents the total amount of heat added to the system during the various processes. It provides insights into the heat input to the system.

2.1.6 Heat Removed (Variable: Scalar)

The `Heat Removed` variable is a scalar that represents the total amount of heat removed from the system during the different processes. It helps in understanding the heat dissipation or removal from the system.

2.1.7 Internal Energy (Variable: Scalar)

The `Internal Energy` variable represents the internal energy of the gas and is calculated based on the specific heat at constant volume (`Cv`). It provides information about the energy stored within the gas.

2.1.8 Temperature (Variable: Vector)

The `Temperature` variable is a vector that records the temperature of the gas over time during the process. Understanding temperature changes is crucial for analyzing thermodynamic behavior.

2.1.9 Pressure (Variable: Vector)

The `Pressure` variable is a vector that tracks changes in pressure over time. It helps in assessing the effects of pressure on the gas during the process.

2.1.10 Volume (Variable: Vector)

The `Volume` variable is a vector that represents the changes in the volume of the gas throughout the process. Analyzing volume changes is essential for understanding the expansion or compression of the gas.

2.1.11 Number of Moles (Variable: Scalar)

The `Number of Moles` variable is a scalar that defines the quantity of gas in moles. It's an essential factor for determining gas properties and behavior.

2.2 Fluid Object Functions

2.2.1 isothermal(P=None, V=None, T=None, n=None)

The `isothermal` function simulates an isothermal process for the Fluid object. You can specify the final pressure (`P`), final volume (`V`), final temperature (`T`), or the number of moles (`n`) as the endpoint of the process. This function changes the state of the gas while keeping the temperature constant.

2.2.2 adiabatic(P=None, V=None, T=None, n=None)

The `adiabatic` function simulates an adiabatic process for the Fluid object. Similar to the `isothermal` function, you can specify the final pressure (`P`), final volume (`V`), final temperature (`T`), or the number of moles (`n`) as the endpoint of the process. Adiabatic processes do not exchange heat with the surroundings.

2.2.3 isobaric(P=None, V=None, T=None, n=None)

The `isobaric` function simulates an isobaric process for the Fluid object. You can specify the final pressure (`P`), final volume (`V`), final temperature (`T`), or the number of moles (`n`) as the endpoint of the process. This function changes the state of the gas while keeping the pressure constant.

2.2.4 isochoric(P=None, V=None, T=None, n=None)

The `isochoric` function simulates an isochoric (constant volume) process for the Fluid object. You can specify the final pressure (`P`), final volume (`V`), final temperature (`T`), or the number of moles (`n`) as the endpoint of the process. This function keeps the volume of the gas constant throughout the process.

2.2.5 plot(Fluid, display=None)

The `plot` function is used to generate graphical plots that visualize the thermodynamic process recorded within the Fluid object. It offers various options for displaying different properties such as pressure, volume, and temperature.

2.2.6 show()

The `show` function is used to display the plots generated by the `plot` function. This function simplifies the process of viewing and analyzing the graphical representations of the thermodynamic process.

In conclusion, the Fluid object in the Termopy library provides a wide range of variables and functions to help you model and analyze thermodynamic processes. By utilizing these variables and functions, you can gain a comprehensive understanding of the behavior of ideal gases under various conditions and during different processes.

In conclusion, the Fluid object in the Termopy library provides a wide range of variables and functions to help you model and analyze thermodynamic processes. By utilizing these variables and functions, you can gain a comprehensive understanding of the behavior of ideal gases under various conditions and during different processes.