The PYroMat User and Developer Handbook

Christopher R. Martin
Associate Professor of Mechanical Engineering
Penn State University

July 24, 2021

Contents

1 Introduction **6**

1.1 The Properties 7

1.1.1 Primary Properties 7

1.1.2 Density, ρ 9

1.1.3 Specific volume, v 10

1.1.4 Temperature, T 10

1.1.5 Pressure, p 11

1.1.6 Internal Energy, e 12

1.1.7 Enthalpy, h 12

1.1.8 Entropy, s 14

1.1.9 Specific Heats, c_p c_v 15

2 Getting started **17**

2.1 Installation 17

2.1.1 Prerequisites 17

2.1.2 `pip` 18

2.1.3 `setuptools` 18

2.2 PYroMat from the command line 18

2.2.1 Importing 18

2.2.2 Finding substances 18

2.2.3 Retrieving substance data 18

2.3 Property interface 18

2.3.1 Property method arguments 18

2.3.2 Default values 18

2.3.3 Working with arrays 18

3 Configuration **19**
3.1 The `pm.config` instance . . . 19
3.2 Configuration files . . . 19

4 Units **20**
4.1 Unit definitions . . . 20
4.2 Setup . . . 22
4.3 Constants . . . 23
4.4 `Conversion` Class . . . 25
4.5 Fundamental Units . . . 26
4.5.1 Time . . . 26
4.5.2 Length . . . 27
4.5.3 Mass and Weight . . . 28
4.5.4 Molar . . . 31
4.5.5 Matter . . . 33
4.5.6 Temperature . . . 34
4.6 Derived Units . . . 37
4.6.1 Force . . . 37
4.6.2 Energy . . . 38
4.6.3 Pressure . . . 39
4.6.4 Volume . . . 41

5 The PYroMat load process **44**
5.1 The class registry module, `reg` . . . 44
5.2 The data module, `dat` . . . 44
5.3 Tools . . . 44

6 Ideal Gases **45**
6.1 Properties of ideal gases . . . 46
6.1.1 Ideal gas law . . . 46
6.1.2 Internal energy . . . 49
6.1.3 Enthalpy . . . 51
6.1.4 Specific heats . . . 51
6.1.5 Entropy and enthalpy revisited . . . 53
6.1.6 Other properties . . . 54
6.1.7 Enthalpy of formation . . . 55
6.1.8 Properties of mixtures . . . 57

6.2 The ideal gas collection 62
6.2.1 The Shomate equation: `ig` 62
6.2.2 The NASA polynomial: `ig2` 64
6.2.3 The ideal gas mixture: `igmix` 67

Nomenclature

These variables and units are restricted to derivations in this document. Because PYroMat has a user-configurable unit system, any of these may be expressed in alternate units.

The following are parameters with units.

Symb.	Units	Description
a	m/s	Speed of sound
c	J/kg/K	Specific heat
e	J/kg	Internal energy
h	J/kg	Enthalpy
k	J/K	Boltzmann constant
M	kg	Mass
N	count	Number of moles
n	m^{-3}	Number density
p	Pa	Pressure
q	$\mathrm{J\ kg}^{-1}$	Normalized heat addition
R	$\mathrm{J\ kg}^{-1}\ \mathrm{K}^{-1}$	Gas constant
R_u	$\mathrm{J\ kmol}^{-1}\ \mathrm{K}^{-1}$	Universal gas const.
s	J/kg/K	Entropy
T	K	Temperature
v	m^3/kg	Specific volume
W	kg/kmol	Molar mass
ρ	$\mathrm{kg/m}^3$	Density

The following are dimensionless parameters.

Symb.	Def.	Description
δ	ρ/ρ_c	ND Density
τ	T_c/T	ND Temperature
γ	c_p/c_v	Sp. heat ratio
π	p/p_c	ND Pressure

The following are subscripts used throughout the document.

Symb.	Description
$_c$	Critical point
$_f$	Formation property (e.g. enthalpy of formation)
$_p$	Constant-pressure (sp.heat)
$_t$	Triple point
$_v$	Constant-volume (sp.heat)
$^\circ$	Property at standard pressure

Special constants used in more fundamental calcuations are defined throughout this document. Many are listed explicitly in Table 4.1 in Chapter 4.

Chapter 1

Introduction

PYroMat is a Python-based package for calculating the thermodynamic properties of fluids. That includes liquids, gases, and plasmas. It is written in pure Python, and the core algorithms only depend on the Numpy package for back-end numerical and array support. This document is intended to serve as a reference manual for interpreting, using, or even writing your own PYroMat data sets.

To that end, we begin with a brief review of the thermodynamic properties PYroMat calculates. For a detailed development of these properties, the reader should consult an introductory text on thermodynamics[1].

In the author's opinion, contemporary undergraduate texts on thermodynamics do the reader a disservice by ignoring the deeply intuitive (and terribly important) underpinnings provided by the kinetic theory of gases. Kinetic theory and thermodynamics have a complicated relationship because they were developed in parallel (often in tragic contention with one another), but the limits imposed by many of the common assumptions (such as idea or perfect gas) cannot be deeply understood without kinetic theory. To the interested reader, the author would recommend Jeans's 1949 introductory text[2]. It is old, but it is

[1] For example, Cengel and Boles, *Thermodynamics*, McGraw Hill. Any edition will do.

[2] James Jeans, *Introduction to the Kinetic Theory of Gases*, Cambridge University Press, 1949.

Table 1.1: Thermodynamic properties, their symbols, and their unit systems

Symbol	In-Code	Units	Description
T	`T`	T	Temperature
p	`p`	P	Pressure
ρ	`d`	M / V	Density
x	`x`	-	Quality
R	`R`	E / M / T	Ideal gas constant
W	`mw`	Ma / Mo	Molecular weight
e	`e`	E / M	Internal energy
h	`h`	E / M	Enthalpy
s	`s`	E / M / T	Entropy
c_p	`cp`	E / M / T	Constant-pressure specific heat
c_v	`cv`	E / M / T	Constant-volume specific heat
γ	`gam`	-	Specific heat ratio

brief, accessible, inexpensive, and approaches the subject only requiring that the reader have a grasp of calculus, geometry, and introductory mechanics.

1.1 The Properties

PYroMat is primarily concerned with the *thermodynamic* properties listed in Table 1.1. There is no single set of units used to express these properties since PYroMat uses a unit-configurable unit system. The unit systems given in Table 1.1 are itemized in Table 1.2. PYroMat 's system of units is discussed in detail in Chapter 4.

1.1.1 Primary Properties

When studying a substance, the theory of thermodynamics can make equal use of any set of properties to discover the others, but in a practical sense, it is almost always prudent to define a pair or more of core properties that are standard for defining the thermodynamic state.

For most problems of engineering relevance, temperature and pres-

Table 1.2: Classes of units, their entry in `pm.config`, and their defaults.

Unit	`config` entry	Default	Description
E	`unit_energy`	kJ	Energy
L	`unit_length`	m	Length
M	`unit_matter`	kg	Matter (molar or mass)
Ma	`unit_mass`	kg	Mass
Mo	`unit_molar`	kmol	Molar
P	`unit_pressure`	bar	Pressure
t	`unit_time`	s	Time
T	`unit_temperature`	K	Temperature
V	`unit_volume`	m^3	Volume

sure (T, p) are the favorite, since both are readily measured, and both are of immediate importance to fluid machinery design. Fortunately, all of the common properties of ideal gases can be conveniently constructed directly from these two properties. Unfortunately, that is not the case with non-ideal fluids.

It is intuitive that when gases are compressed into tighter and tighter spaces, the distance between the molecules becomes a vitally important parameter for predicting the substance's properties. Pressure is not a convenient metric of that distance, but density is. For this reason, nearly all non-ideal gas properties are modeled in terms of (T, ρ), and pressure has to be calculated indirectly. This creates a substantial amount of work for the design engineer who may not wish to delve into the details of how properties are calculated.

To address the issue, PYroMat designates four *primary properties* that may be used interchangeably in pairs to specify the thermodynamic state. When more than two are included, the state is overspecified, and PYroMat assumes that the values specify a consistent state. Which two are used to calculations are selected in order of precedence shown in Table 1.3.

For example, let's say I needed to calculate the specific heat of oxygen.

```
>>> import pyromat as pm
>>> o2 = pm.get('ig.O2')
```

Table 1.3: Primary properties and their order of precedence

Order	Symb.	Description
1	T	Temperature
2	p	Pressure
3	ρ	Density
4	x	Quality

```
>>> o2.cp(300,2)
```

How should the two numbers (300,2) be interpreted? Using the order of precedence, those refer to a state $T = 300$ K and $p = 2$ bar using PYroMat's default units. To deliberately specify density, a keyword syntax should be used instead.

```
>>> o2.cp(d=1.2, p=2)
```

1.1.2 Density, ρ

It is convenient to begin our discussion of properties with density since it is the easiest to define.

Density is the quantity of a substance per unit volume it occupies in space. It can be described as a number of molecules or mass per unit space. When described with molar units, it is usually called concentration, but PYroMat does not make that distinction.

When the unit volume is very tiny, this is a poorly defined quantity. For example, as the volume shrinks to be about the same size as the distance between molecules, the density one might measure would vary hugely as individual molecules entered and left the region of space. However, as the volume grows within a thermodynamically homogeneous region, the ratio of matter to volume converges to a consistent well-defined value.

This introduces the idea that the study of thermodynamics is essentially a careful study of averages over a large population of mechanical bodies. It is important that we study a quantity of a substance large enough that the extremes of individual molecules do not weigh heavily in our measurements. On the other hand, our measurements must consider a region of space small enough that the properties that interest

us do not vary significantly. We may consider a region to be thermodynamically homogeneous if it can be divided into two equal smaller regions that exhibit identical thermodynamic properties.

The density, in molar units, is

$$\langle \rho \rangle = \frac{N}{V}, \tag{1.1}$$

where V is the volume of the region, and N is the number of molecules. The number of molecules may be expressed either in a literal count or in molar units (See Section **??**). In mass units, the density is merely multiplied by the molecular weight of the species,

$$\rho = W \frac{N}{V}. \tag{1.2}$$

Since ρ is not available in the ASCII character set, PYroMat uses `d` to represent density.

1.1.3 Specific volume, v

Specific volume is the volume occupied by a unit of matter, and is calculated as the inverse of density.

$$v \equiv \frac{1}{\rho}. \tag{1.3}$$

Even though density is the property with the clearer fundamental definition, specific volume is mathematically identical to other intensive properties because it is formulated per unit matter.

PYroMat does not calculate specific volume directly, but deals in density instead. The user is called on to calculate `v=1/d` when specific volume is needed.

1.1.4 Temperature, T

Temperature scales were originally developed as quantitative means for describing hot and cold, but they were developed before we had more physical descriptions for their meaning. After all, the existence

of molecules and atoms was still being hotly debated while temperature scales were already in wide scientific and engineering use.

We now understand temperature to be an observable measure of the molecular translational kinetic energy of a substance. In a gas, the molecules are free to translate through space, and temperature is proportional to their kinetic energy. In a liquid or solid, the same energy manifests as molecular vibration within the confines of the intermolecular forces.

For a gas,

$$\langle \frac{1}{2} m u^2 \rangle = \frac{3}{2} kT, \tag{1.4}$$

Here, m is the mass of an individual molecule, $\langle u^2 \rangle$ is the mean square of velocity, k is the Boltzmann constant, and T is the temperature in absolute units.

The ITS-90 temperature scale establishes an international standard for the definition of temperature in terms of the triple points of various pure substances. Many of the property models included in PYroMat were formulated when the previous ITPS-68 was the international temperature scale, but the changes were so minute that the uncertainties in the properties dominate[3]. As a result, they are treated as interchangeable for the purposes of this handbook.

1.1.5 Pressure, p

Pressure is the static force exerted by a fluid on a surface. It is quantified in force per unit area of the surface, and it always acts normal to the surface facing into the surface (away from the fluid).

In a gas, pressure is due to the impact of molecules on the surface. Pressure may be increased by increasing either their velocity (temperature) or their quantity (density). Because these effects are linear in a gas, this leads to the famous ideal gas relationship between density, temperature, and pressure.

In a liquid or solid, intermolecular forces that cause pressure are persistent instead of momentary (due to collisions in a gas). Under

[3] Page 10 of Part I of Malcom Chase, *NIST-JANAF Thermochemical Tables*, Journal of Physical and Chemical Reference Data, Monograph 9, 1998.

these conditions, even slight changes in intermolecular spacing causes huge changes in pressure, making the substance quite stiff in comparison to gases. In this case as well, increasing temperature makes molecules vibrate more violently in their equilibrium with each other, so at a consistent average density, the force applied to a neighboring surface will increase. This is why substances appear to expand as they are heated.

The standard atmosphere is defined as the global mean pressure at sea level, and is defined to be 101,325Pa.

1.1.6 Internal Energy, e

Energy can be stored in a fluid in many ways. In gases, for example, the molecules translate with great speed (see temperature), the molecules vibrate and rotate, and there is incredible energy stored in the chemical bonds of molecules. However, all of these are dwarfed by the energy contained in the forces binding the nucleus of each atom.

We account for *all* of these energies simultaneously with the internal energy, e, which has units energy per matter (e.g. J/kg). It is neither practical nor necessary to tally all of these energies in an absolute fashion. Especially since most applications will have no release of nuclear energy, it is practical, instead, to describe how the substance's energy changes relative to some reference state. This is not unlike defining a reference height for gravitational potential energy calculations in classical mechanics. The choice is arbitrary and has no bearing on the result, but it can drastically simplify the calculations.

Therefore, we say that the internal energy, e is the sum of vibrational, rotational, translational, chemical, and nuclear energies contained in a thermodynamically homogeneous unit matter, subtracted by the same sum at some reference state.

1.1.7 Enthalpy, h

When a fluid of any kind is flowing, it caries its internal energy with it, but it also does mechanical work as it flows. The mechanical work done by a moving fluid is $p\mathrm{d}V$, where p is the pressure exerted and $\mathrm{d}V$ is a differential volume displaced by the fluid. If we were to imagine that the

volume were displaced while the fluid is expanding or contracting, the same idea applies to a fluid whether it is flowing or not. Per unit mass, this can be expressed as $p\mathrm{d}v$. Integrated over an isobaric (constant-pressure) process, this becomes simply pv.

It is a matter of convenience for engineers and physicists that deal with fluid power, that we define enthalpy as the sum of internal energy and fluid power,

$$h \equiv e + pv = e + \frac{p}{\rho}. \tag{1.5}$$

This term appears naturally when energy balances are applied to open systems (ones where flow is moving through the system). Internal energy accounts for all of the energy stored by the molecules in a fluid, and enthalpy additionally includes the fluid's capacity to do mechanical work as it flows.

Enthalpy is a most commonly used in the analysis of any flowing fluid such as in heat exchangers, combustors and burners, chemical reactors, compressors, turbines, valves, etc... Even though it is derived from a property that might be argued to be more "fundamental," enthalpy is usually tabulated as a primary property because it is so useful.

Just like internal energy, a substance's enthalpy is defined relative to an arbitrary reference state. However, since internal energy and enthalpy share a common definition, their reference states *must* be consistent with one another. Furthermore, for any data set intended for the analysis of chemical reactions, the enthalpies of chemically related species (such as H_2, O_2, and H_2O) *must* be defined consistently so that the changes in molecule internal energy due to chemical reaction can be properly accounted for. As a result, there are certain *reference species* for which arbitrary reference state enthalpies are selected.

Since the PYroMat multi-phase models permit only pure substances, they do not need rigorously defined reference states. However, the ideal gas models are defined with this in mind to permit chemical action. See the ideal gas chapter (Chapter 6) for a detailed discussion on the enthalpy of formation.

1.1.8 Entropy, s

The idea of entropy is born with the Clausius statement of the Second Law of Thermodynamics, which says that a reversible cyclic series of processes that include heat transfer must obey

$$\oint \frac{\delta q}{T} \underset{rev.}{=} 0, \tag{1.6}$$

where T is the temperature of the substance and δq is the addition of heat. It is important to emphasize that this is only true of processes that result in a continuous cycle where the fluid ends at the same thermodynamic state from which it began (like in an engine or refrigeration cycle).

The first law would be satisfied by any cycle where the work and heat transfer merely summed to zero, but the Clausius equality implies something deeper. Heat is not a property of the substance, but heat added reversibly in ratio with the temperature consistently returns to zero when the substance returns to its original thermodynamic state. That implies the existence of a new (and terribly important) property. Clausius termed that property entropy,

$$\mathrm{d}s \underset{\text{rev.}}{\equiv} \frac{\delta q}{T} \tag{1.7}$$

From the first law of thermodynamics, $\delta q = \mathrm{d}e + p\mathrm{d}v$, and we have a way to calculate entropy from the other properties

$$T\mathrm{d}s = \mathrm{d}e + p\mathrm{d}v \tag{1.8}$$

$$= \mathrm{d}h - v\mathrm{d}p \tag{1.9}$$

Just like with temperature, there is a parallel (mathematically consistent) definition of entropy from statistical mechanics. Entropy can also be understood as the probability of the substance obtaining a thermodynamically equivalent state. The probability of gas molecules spontaneously exhibiting a specific set of positions and velocities is astronomically low, but there is an infinity of similar positions and velocities that would give the gas precisely the same temperature (for

example). The probability of gas molecules marching orderly in a lattice like a crystal is also astronomically low, but this condition has very few alternatives, making it truly unlikely.

Just like internal energy, it is theoretically possible to tally all of the possible states and calculate the probability of each, but that task is neither practical nor necessary. Instead, it is convenient to define the entropy as zero at some reference state and then deal merely in changes in entropy as defined by (1.7).

1.1.9 Specific Heats, c_p c_v

Of the properties so far defined, only temperature, pressure, and density can be directly measured. The specific heats are vitally important to the study of a substance, because they can be conveniently directly measured and the other properties calculated from them.

Specific heat, c, is the amount of heat per mass of a substance, δq, required to obtain a small increase in temperature, δT,

$$c \equiv \frac{\delta q}{\delta T}. \tag{1.10}$$

Especially when dealing with gases, we have to be more specific because substances have a tendency to expand when they are heated. One measures a different value for specific heat depending on whether or not expansion is allowed.

For any process, energy will be conserved, so

$$\delta q = \mathrm{d}e + p\mathrm{d}v. \tag{1.11}$$

Here, δq is heat added per mass of the substance, $\mathrm{d}e$ is the change in internal energy, p is the pressure, and $\mathrm{d}v$ is the change in volume per unit mass.

When heat is applied in such a manner that the substance's volume is constant, $\mathrm{d}v = 0$, the mechanical work is zero, and all of the heat is stored as internal energy.

$$\begin{aligned} \delta q|_v &= \mathrm{d}e \\ &= \left(\frac{\partial e}{\partial T}\right)_v \mathrm{d}T \end{aligned} \tag{1.12}$$

Thus, the constant-volume specific heat is, by definition,

$$c_v \equiv \left(\frac{\delta q}{\delta T}\right)_v = \left(\frac{\partial e}{\partial T}\right)_v . \tag{1.13}$$

In this process, the substance's pressure will rise (sometimes sharply) as it is heated in a rigid container.

When one considers addition of heat under constant pressure, a trick application of the chain rule for the term, pv, lets us transition the differential on volume into a differential on pressure. The definition for enthalpy (**??**) appears naturally.

$$\begin{aligned} \delta q &= \mathrm{d}e + p\mathrm{d}v + v\mathrm{d}p - v\mathrm{d}p \\ &= \mathrm{d}(e + pv) - v\mathrm{d}p \end{aligned} \tag{1.14}$$

When heat is added while pressure is constant (like in an atmospheric gas), $\mathrm{d}p = 0$, and

$$\begin{aligned} \delta q|_p &= \mathrm{d}h \\ &= \left(\frac{\partial h}{\partial T}\right)_p \mathrm{d}T \end{aligned} \tag{1.15}$$

Thus, constant-pressure specific heat is, by definition,

$$c_p \equiv \left(\frac{\delta q}{\delta T}\right)_p = \left(\frac{\partial h}{\partial T}\right)_p . \tag{1.16}$$

Chapter 2

Getting started

In this chapter, we discuss everything you should need to get your own installation of PYroMat working and providing properties.

2.1 Installation

PYroMat is distributed primarily through the Python package index (https://pypi.org/), but it can also be downloaded and installed "manually." PYroMat is written in plain Python, so there is no need to compile to binaries.

2.1.1 Prerequisites

PYroMat requires NumPy (https://numpy.org/) version 1.7 or later for numerical and array support. There are no other prerequisites. For documentation on the custom back-end numerical routines implemented in PYroMat, see chapter **??**.

2.1.2 `pip`

2.1.3 `setuptools`

2.2 PYroMat from the command line

2.2.1 Importing

2.2.2 Finding substances

2.2.3 Retrieving substance data

2.3 Property interface

2.3.1 Property method arguments

2.3.2 Default values

2.3.3 Working with arrays

Chapter 3

Configuration

3.1 The `pm.config` instance

3.2 Configuration files

Chapter 4

Units

PYroMat handles a wide variety of units automatically, but interested users are also welcome to use the back-end algorithms for their own purposes. All of the core substance methods deal in their own native units, but the `pyromat.units` module contains the tools that are responsible for converting to and from the units specified in the `pyromat.config` system (see Chapter **??**).

A list of all currently available units is available by typing

```
>>> import pyromat as pm
>>> help(pm.units)
```

To obtain information specific to one of the unit classes (length, for example), type

```
>>> help(pm.units.length)
```

The unit systems used by the property methods and their corresponding configuration entries are listed in Tables 1.1 and 1.2.

This chapter infuses a little history along with the information PYroMat users are likely to find useful when interacting with the unit conversion system.

4.1 Unit definitions

The Bureau International des Poids et Mesures (BIPM) International Committee for Weights and Measures (CIPM) is the entity responsible

for the definition of the fundamental units on which the SI system is based. Nearly all other units in wide use are now derived from the SI units in some way, so rigorous treatment of their formal definition is worth some attention. There are countless web-based unit conversion calculators, but few use sufficient care in the derivation of their conversion factors to be trusted for serious work.

The second (time), meter (length), kilogram (mass), ampere (electrical current), kelvin (temperature), mole (quantity), and candela (luminous intensity) are the fundamental units in terms of which all other measures are derived. Contemporary definitions for these units are constructed so that certain important physical constants (like Plank's or Boltzmann's constants) may be represented exactly with finite precision, so the conversion factors used in PYroMat are, by definition, exact.

Before the turn of the twentieth century, the United States and the British governments had adopted the international yard and avoirdupois pound. This move re-defined the pound and the yard (and all those based on them) in terms of the meter and kilogram using an exact relationship with finite precision. The contemporary definitions for those fundamental units are constructed from highly stable natural phenomena may be independently reproduced in a laboratory, but this idea is quite recent. Abandoning the tradition of defining units from a standard bar or mass did not happen until the latter half of the twentieth century. [**?**]

This establishes a web of precise definitions of units for US customary, imperial, and SI systems. However, there are still popular measures (like the inch water column for pressure or scf for quantity of a gas) that depend on establishing so-called "standard" conditions whose precise definitions are often neglected (and are far from standard).

For example, water and mercury column measures for pressure depend on the density of a fluid and the local strength of gravity, both of which are subject to change. So-called "standard" pressure is often 1atm (1.01325bar), but is precisely 1bar for the NIST-JANAF data. Worse still, standard temperature usually refers to 273.15K (0°C), but the NIST-JANAF tables use 298.15K (25°C), and there are some engineering applications that still use 70°F. There are at least a half-dozen

definitions for the calorie, which are construed from the specific heat of water at different conditions. These inconsistencies are far too severe to ignore, so great care is required when pulling data from multiple sources into a single database.

4.2 Setup

To maintain transparency in regard to the standards used to derive unit conversions, the PYroMat units system includes a `setup` function that is responsible for constructing all of the unit conversion routines. Users can inspect the constants it defines, and users may change them at any time by re-calling the `setup` function with new arguments. Its behavior is fully documented and can be found by evoking `help` as below.

```
>>> import pyromat as pm
>>> pm.units.setup(Tstd=273.15, pstd=1.01325, \
...     g=9.80665, dh2o = 999.972, dhg=13595.1)
>>> help(pm.units.setup)
```

It should be emphasized that there is a difference between the standard parameters used to construct the PYroMat unit system and the default parameters determined by `pm.config`. The standard conditions are only used to construct the unit conversion system, they are always given in the units documented below, and they are ignored outside of the `units` module. On the other hand, the default parameters are used in property methods when arguments are omitted, and they are interpreted in whatever units are currently configured. See the configuration chapter (Chapter **??**) for more information.

Standard temperature and pressure, `Tstd` and `pstd`, *must* be entered in units of Kelvin and bar respectively (regardless of the units configured in `pm.config`). The standard atmosphere was set to precisely 1.013 25 bar by the BIPM General Conference on Weights and Measures (CGPM) in 1901 [**?**], and has been almost universally adopted. The default standard temperature is less universal, but is usually 273.15 K, the freezing point of water at one atmosphere of pressure.

The CGPM did not establish a standard acceleration due to gravity

for the purpose of weights and measures until 1954. It is set to be precisely 9.80665 m s^{-2} [**?**]. This is a mean of gravitational forces experienced at sea level at a latitude of 45 degrees. This value can be found to agree precisely with the official conversions for pounds mass, pounds force, kilogram, and newton.

The densities of water and mercury, `dh2o` and `dhg`, must be entered in units of kg m^{-3}. The default for water is its density at atmospheric pressure and 4°C. The default for mercury is its density at atmospheric pressure and 0°C.

4.3 Constants

The `setup` function is called when the `units` module is first imported to initialize the PYroMat unit constants listed in Table 4.1 and declare all the unit conversion routines. These constants are used widely throughout the PYroMat system, so altering them is only recommended for advanced users who can anticipate the effects they will have.

Regardless of experience, constants should never be changed manually. They should only ever be changed be re-calling the `setup` function since the unit conversion routines will need to be updated to reflect the new value.

The values in this table are, by the definitions of units, exact (unless noted with ≈). Similarly, the conversions in the following sections are also exact unless they are otherwise noted.

Table 4.1: PYroMat constants and their default values. Values with a * are affected by calls to `setup`.

Sym.	Name	Value		Description
g^*	`const_g`	9.806 65	m s^{-2}	Gravity at 45° lat.
h	`const_h`	6.626 070 15$\times 10^{-34}$	J s	Plank constant
k	`const_k`	1.380 649 $\times 10^{-23}$	J K^{-1}	Boltzmann constant
N_a	`const_Na`	6.022 140 857$\times 10^{23}$	mol^{-1}	Avagadro number
N_c	`const_Nc`	6.241 509 34$\times 10^{18}$	C^{-1}	Electrons per Coulumb
$\overline{\rho}_{std}{}^*$	`const_nstd`	44.615 048 197 83	mol m^{-3}	Standard molar density
$p_{std}{}^*$	`const_pstd`	1.013 25	bar	Standard pressure
q	`const_q`	1.602 176 634 $\times 10^{-19}$	C	Fundamental charge
R_u	`const_Ru`	$\approx$ 8.314 462 618	J mol^{-1} K^{-1}	Universal gas constant
$T_{std}{}^*$	`const_Tstd`	273.15	K	Standard temperature
$\rho_{H_2O}{}^*$	`const_dh2o`	999.972	kg m^{-3}	Std. density of water
$\rho_{Hg}{}^*$	`const_dhg`	13595.1	kg m^{-3}	Std. density of mercury

4.4 `Conversion` Class

Unit conversions are performed by a callable `Conversion` class. With the exception of the `matter` and `temperature_scale` functions (described below), each of the conversions is performed by a callable `Conversion` class instance defined by the `setup` function at import.

As a callable, `Conversion` instances mimic a function that accepts up to five argument,

```
>>> conversion_instance(value=1., \
...     from_units=None, to_units=None,\
...     exponent=None, inplace=None)
```

The value may be a scalar or a numpy array of values to be converted. By default, it is 1, so that if no value is specified, the result will automatically be a conversion factor between the given units.

The `from_units` and `to_units` keywords accept strings identifying the units in the conversion. In the event that one of them is omitted, each `Conversion` instance is initialized to check for an entry establishing an default unit in the `pm.config` system. In the sections that follow, each of the unit classes is listed with the name of its instance, the configuration entry that identifies its default unit, and the default that is configured in PYroMat at installation.

The `exponent` keyword allows for powers of units that are not unity. For example, units of velocity are not explicitly included, but it is still easy to convert from meters per second to miles per hour by

```
>>> import pyromat as pm
>>> speed_ms = 10.
>>> temp = pm.units.length(speed_ms, 'm', 'mile')
>>> speed_mph = pm.units.time(temp, 's', 'hr', \
...     exponent=-1)
>>> print(speed_mph)
22.369362920544024
```

Most users will never need the `inplace` keyword, but when it is set to `True` and the `value` is a numpy array, the original values will be overwritten with the result of the conversion.

`Conversion` instances also support item recall, so that the conversion factor for any unit may be obtained simply

```
>>> print(pm.units.volume['m3'] / \
...     pm.units.volume['L'])
1000.
```

This indicates that there are 1000 cubic meters per liter. Most users will not need to use this capability, but for those that do, it is important to always use ratios of units of the same class. Each item recall returns a scalar value indicating the *relative* size of the respective unit, but relative to what? In the case of volume, each value indicates the unit's value in cubic meters, but that is an arbitrary choice.

For a complete list of the supported units in a unit class, use the `get()` method.

```
>>> pm.units.energy.get()
dict_keys(['J', 'kJ', 'cal', 'kcal', 'BTU_ISO',
    'eV', 'BTU'])
```

4.5 Fundamental Units

Fundamental units are those that have precise definitions based on observable phenomena in nature.

4.5.1 Time

`Conversion` instance:	`pm.units.time`
`pyromat.config` entry:	`'unit_time'`
Default:	`'s'`

To alter the PYroMat unit for time,

```
>>> import pyormat as pm
>>> pm.config['unit_time'] = 'min'
```

Time is the most fundamental measure defined by the BIPM. "[The second] is defined by taking the fixed numerical value of the caesium frequency $\Delta\nu_{\mathrm{Cs}}$, the unperturbed ground-state hyperfine transition frequency of the caesium-133 atom, to be 9,192,631,770 when expressed in

Table 4.2: Time units recognized by PYroMat

Setting	Value	Description
`year`	31,536,000 s	year
`day`	86,400 s	day
`hr`	3,600 s	hour
`min`	60 s	minute
`s`	1 s	second
`ms`	0.001 s	millisecond
`us`	10^{-6} s	microsecond
`ns`	10^{-9} s	nanosecond

the unit Hz, which is equal to s^{-1}."[?] Most of the other fundamental units are constructed in terms of it.

The other units of time recognized by the PYroMat system are well known, and have simple definitions in terms of the second. It is worth emphasizing that the year and the day recommended by NIST [?] presume precisely 24 hours to a day and 365 days to a year. This ignores the 0.25 year annual error accounted for by leap years.

Their configuration strings, their values in terms of the second, and their names are listed in Table 4.2.

Time is implemented in the unit conversion system for completeness, but as of PYroMat version 2.1.0, it is not used directly by any of the property methods.

Listing 4.1: Time Conversion Example

```
>>> pm.units.time(30, 's', 'min')
0.5
```

4.5.2 Length

`Conversion` instance:	`pm.units.length`
`pyromat.config` entry:	`unit_length`
Default:	`'m'`

To alter the PYroMat unit for length,

```
>>> import pyormat as pm
>>> pm.config['unit_length'] = 'in'
```

"[The meter] is defined by taking the fixed numerical value of the speed of light in vacuum, c, to be 299,792,458 when expressed in the unit m s^{-1} [...]."[**?**] All other metric length units are trivial to derive from the meter.

The international yard (and by extension, the inch, foot, and mile) were defined exactly in terms of the meter before the end of the nineteenth century, but their contemporary values were not adopted until the middle of the twentieth century.[**?**] Now one yard is defined to be precisely 0.914 4 meters, which results in the better known relationship, one inch is precisely 25.4 millimeters.

The meter was once defined to be a fixed ratio of a great circle around the Earth. This made it of great use for navigation, and for the same reason, the nautical mile is still in common use. The meter was one ten millionth of one quarter of a meridian, making the circumference of the Earth precisely 40,000 km by definition [**?**]. The nautical mile was was also primarily used for navigation, but it was set to the distance covered by one arc minute of latitude (of which there are 21,600 in a circle). Therefore, the nautical mile was exactly 40,000,000 m / 21,600 or approximately 1,851.851 85 m. In the middle of the twentieth century, the United States Departments of Commerce and Defense jointly declared the nautical mile to be redefined as precisely 1,852m [**?**].

The length scales supported by PYroMat and their values in terms of the meter are listed in table 4.3.

Example:

```
>>> pm.units.length(12, 'in', 'm')
.127
```

4.5.3 Mass and Weight

`Conversion` instance:	`pm.units.mass`
`pyromat.config` entry:	`unit_mass`
Default:	`'kg'`

Table 4.3: Length units recognized by PYroMat

Setting	Value	Description
`km`	1,000 m	kilometer
`m`	1 m	meter
`cm`	0.01 m	centimeter
`mm`	0.001 m	millimeter
`um`	1×10^{-6} m	micrometer
`nm`	1×10^{-9} m	nanometer
`A`	1×10^{-10} m	Angstrom
`in`	0.025 4 m	inch
`ft`	0.304 8 m	foot
`yd`	0.914 4 m	yard
`mile`	1,609.344 m	statute mile
`mi`	Alternate of `mile`	
`nmi`	1,852 m	nautical mile

To alter the PYroMat unit for mass,

```
>>> import pyormat as pm
>>> pm.config['unit_mass'] = 'lb'
```

"[The kilogram] is defined by taking the fixed numerical value of the Planck constant h to be $6.62607015 \times 10^{-34}$ when expressed in the unit J s, which is equal to kg m^2 s^{-1}"[**?**]. Like many of the remaining fundamental units, the kilogram is defined in terms of length and time, making it dependent on their definitions as well.

The atomic mass unit (u or amu) is defined as precisely 1/12 the mass of a carbon 12 atom at rest. However, in PYroMat the value of one u is calculated in kilograms by enforcing that 1000 moles of a substance with 1u of mass will have 1kg of mass. These two definitions of the atomic mass unit are equivalent to the precision of their definition[**?**, p.209].

Definitions for the pound and units derived from it are often confused by conflicting definitions of the term "weight." For example, in NIST special publications it is possible to find "[...] the weight of a body in a particular reference frame is defined as the force that gives the body an acceleration equal to the local acceleration of free fall in

Table 4.4: Mass units recognized by PYroMat

Setting	Value	Description
`kg`	1 kg	kilogram
`g`	0.001 kg	gram
`mg`	1×10^{-6} kg	milligram
`lbm`	0.453 592 37 kg	pound-mass
`lb`	Alternate form of `lbm`	
`oz`	0.028 349 523 125 kg	ounce
`slug`	$\approx$ 14.593 902 9 kg	slug
`u`	$\approx 1.660\ 539\ 06 \times 10^{-27}$ kg	atomic mass unit
`amu`	Alternate form of `u`	

that reference frame"[**?**, p.23] and "In general usage, the term 'weight' nearly always means mass, and this is the meaning given the term in U.S. laws and regulations." [**?**, p.10]

However, at the end of the nineteenth century, the definition of the avoirdupois pound was set to a fixed fraction of the kilogram, formalizing the definition of the pound as a measure of mass and not force. The distinction became important as the precision of measuring instruments exceeded the consistency of the strength of gravity over the Earth's surface. The contemporary value for the avoirdupois pound was established in the middle of the twentieth century, and is precisely 0.453 592 37 kg of mass.[**?**]

The troy pound and the troy ounce predate the avoirdupois pound as a measure primarily used for quantities of precious metal in coinage [**?**, p.6], but today they are rarely used outside of these specialized applications. They are omitted from PYroMat to avoid confusion and because they are not often used in scientific or engineering applications.

Table 4.4 shows the mass units recognized by PYroMat and their values in kilograms. Note that only the atomic mass unit and the slug have been rounded. The rest of the relationships are precise by definition.

Example:

```
>>> pm.units.mass(2, 'lb', 'kg')
0.90718474
```

4.5.4 Molar

`Conversion` instance:	`pm.units.molar`
`pyromat.config` entry:	`unit_molar`
Default:	`'kmol'`

To alter the PYroMat unit for molar quantities,

```
>>> import pyormat as pm
>>> pm.config['unit_molar'] = 'lbmol'
```

"One mole contains exactly 6.022 140 76 $\times 10^{23}$ elementary entities. This number is the fixed numerical value of the Avogadro constant, N_a, when expressed in the unit mol^{-1} and is called the Avogadro number."[**?**]. Because a mole is a unit of counting, it is equally valid to refer to a mole of planets (about right for our observable universe) as it is to refer to a mole of gas.

The mole is the quantity of a substance with a total mass in grams numerically equal to the mass of the elementary entities expressed in atomic mass units (see Section 4.5.3). This relationship is mirrored by more recently invented alternatives such as the kilogram-mole (kmol) or the pound-mole (lbmol), which enjoy the same relationships with their mass-based namesakes.

If the mass of a single molecule is m_0, a mass, m, of many such molecules must contain

$$N = \frac{m}{m_0} \tag{4.1}$$

molecules.

By definition, when $N = N_a$, the mass of the group expressed in grams will equal the mass of the molecule expressed in atomic mass units. Therefore, the gram-mole is

$$N_{\mathrm{g}} = N_a = \frac{1\mathrm{g}}{1\mathrm{u}}. \tag{4.2}$$

The same process may be applied for any other unit of mass, so the quantity required for m to be expressed in kilograms is

$$N_{\mathrm{kg}} = \frac{1\mathrm{kg}}{1\mathrm{u}}. \tag{4.3}$$

Therefore,

$$\frac{N_{\mathrm{kg}}}{N_{\mathrm{g}}} = \frac{1\mathrm{kg}}{1\mathrm{g}} = 1000. \tag{4.4}$$

The moles are converted by the same conversion factors as their mass equivalents, so a kilogram-mole is precisely 1000 times the gram-mole. To express a quantity, N, in molar units, its quantity only needs to be divided by the quantity of the corresponding mole type.

For describing quantities of a gas, standard (or normal) volumetric units are used so widely that the problematic aspects of their definitions are worth tolerating. A standard (or normal) volume is the quantity of an ideal gas that would occupy that volume at standard conditions. US Customary and imperial units use the word "standard" and metric units use the word "normal," but the meaning is the same.

Regardless of the mass of the elementary particles, an ideal gas has a consistent number density at given conditions,

$$n_{std} = \frac{p_{std}}{kT_{std}}, \tag{4.5}$$

given in total number of molecules per unit volume. Obviously, precise and consistent definitions for standard conditions are essential here. By default PYroMat uses $p_{std} = 1.01325$ bar and $T_{std} = 273.15$ K. When n_{std} is expressed in gram-moles, it is approximately 44.615 033 4 moles per cubic meter.

To calculate the number of moles in a standard volume, one need only multiply by the volume in question. So, a normal liter contains .044 615 033 4 moles (when standard temperature and pressure are as above).

Table 4.5 shows the molar units and their values expressed in kilogram-moles. The values marked with * are dependent on the standard conditions provided when `setup()` was last called.

Even though the mole is the BIPM standard for quantity of a substance, PYroMat uses the kmol or kilogram-mole as the default molar unit for self consistency. For example, the molecular weight property methods return mass per molar units. If the units were set to

Example:

Table 4.5: Molar units recognized by PYroMat

Setting	Value	Description
`kmol`	1 kmol	kilogram-mole
`mol`	0.001 kmol	gram-mole
`lbmol`	0.453 592 37 kmol	pound-mole
`n`	$\approx 1.660\ 539\ 06 \times 10^{-27}$ kmol	count
`Nm3`*	$\approx 0.044\ 615\ 033\ 4$ kmol	normal cubic meters
`Ncum`*	Alternate form of `Nm3`	
`NL`*	$\approx 44.615\ 033\ 4 \times 10^{-6}$ kmol	normal liters
`Ncc`*	$\approx 44.615\ 033\ 4 \times 10^{-9}$ kmol	normal cubic centimeters
`scf`*	$\approx 0.001\ 263\ 357\ 06$ kmol	standard cubic feet
`sci`*	$\approx 0.731\ 109\ 408 \times 10^{-6}$ kmol	standard cubic inches

```
>>> pm.units.molar(1.5, 'scf', 'kmol')
0.0018950355849594869
```

4.5.5 Matter

Function:	`pm.units.matter`
`pyromat.config` entry:	`unit_matter`
Default:	`'kg'`

To alter the PYroMat unit for matter quantities,

```
>>> import pyormat as pm
>>> pm.config['unit_matter'] = 'lbm'
```

Molar and mass units provide parallel methods for quantifying amounts of matter. Thermodynamic properties can be expressed in either, so PYroMat uses a third class of units, *matter*, which is an amalgamation of all molar and mass units. A unit matter may be any of the units listed in Tables 4.4 and 4.5. Since every molar unit can be related to the kilogram-mole and every mass unit can be related to the kilogram, it is only important that we establish how the kilogram is related to the kilogram-mole.

The ratio between kilograms and kilogram moles is a property of the molecule, and is called its molar or molecular weight, W. It is

typically expressed in terms of atomic mass units per molecule, but it is the same in kilograms per kilogram moles. The mass, m of a molar quantity, N, then, is

$$m = NW. \tag{4.6}$$

Unlike any of the other unit conversions, this one depends on the properties of the substance itself. As a result, it is implemented in a custom function that adds a mandatory molecular weight (in u per molecule), `mw`, argument to the other standard `Conversion` instance arguments.

```
>>> matter(value, mw, from_units=None, \
... to_units=None, exponent=None, inplace=False)
```

If the to- and from-units are in the same molar or mass class, then the function merely calls the appropriate `Conversion` instance.

These examples consider a molecule with molecular weight exactly 2.0 u (H_2):

```
>>> import pyromat as pm
>>> pm.units.matter(1, 2, 'kg', 'kmol')
0.5
>>> pm.units.matter(1, 2, 'kg, 'lbmol')
1.1023113109243878
>>> pm.units.matter(1,2,'kg','lb')
2.2046226218487757
>>> pm.units.mass(1, 'kg', 'lb')
2.2046226218487757
```

4.5.6 Temperature

`Conversion` instance:	`pm.units.temperature`
Scale function:	`pm.units.temperature_scale`
`pyromat.config` entry:	`unit_temperature`
Default:	`'K'`

To alter the PYroMat unit for temperature quantities,

```
>>> import pyormat as pm
>>> pm.config['unit_temperature'] = 'K'
```

This history of temperature as a measure of hot and cold extends back longer than thermodynamics as a rigorous theory. Without knowing what underlying principles caused substances to be hot or cold, a Celsius scale could still be reliably be marked onto a thermometer by marking its readings in ice water and boiling water and by dividing the space between into one hundred equal increments.

With the discovery that temperature indicates the mean translational kinetic energy of the molecules of an ideal gas, the kinetic theory of gasses gives the relationship,

$$\frac{1}{2} m_0 \left\langle u^2 \right\rangle = \frac{3}{2} kT. \tag{4.7}$$

The coefficient, k, which is now known as Boltzmann's constant, establishes the proportionality between kinetic energy and temperature. This also provides the idea of an *absolute* temperature scale; one in which zero temperature corresponds to zero energy rather than an arbitrary choice (like the freezing point of water at standard pressure). In this way, Boltzmann's constant completely determines an absolute temperature scale in terms of the other units.

The BIPM stipulates that "[the kelvin] is defined by taking the fixed numerical value of the Boltzmann constant k to be 1.380649×10^{-23} when expressed in the unit J K^{-1}, which is equal to kg m^2 s^{-2} K^{-1} [...]"[**?**] This makes the value for the Boltzmann constant exact by definition.

The realization that temperature is a measure of certain portions of a system's energy is especially useful in plasma physics and related fields. In these studies, it is useful to express temperature directly as an energy, or electron-volts (eV). The temperature of a substance expressed in electron-volts is the electrical potential that can be built up due to thermal motions of electrons, and is equal to kT/q, when q is the fundamental charge.

The rankine scale is the US customary and imperial equivalent to the kelvin, with one 1.8 R per 1 K. The Celsius scale uses the same increments as the kelvin scale, but its zero is set 273.15 K, the freezing

point of water at 1 atmosphere. The Fahrenheit scale uses the same increment as the rankine, but its zero is set so that the freezing point of water at standard pressure occurs at precisely 32°F.

Scales with offsets defy the unit conversion rules implemented by the `Conversion` instance; that all measurements can be converted between the units merely by multiplying by a factor. Obviously, when converting values for a temperature, it is important to handle these offsets correctly, but when considering derivatives involving temperature or other changes in temperature, the offsets must be ignored. As a result, there are two temperature-based unit conversion tools.

The `pm.units.temperature` is a `Conversion` instance should only be used in cases where changes or derivatives in temperature are being considered (e.g. in specific heat or entropy). In these cases, offsets can be ignored, and the conversion factor rules are observed.

When temperatures are being converted between scales so that the offsets between them must be respected, the `pm.units.temperature_scale` function should be used instead. It is a function that mimics the `Conversion` call signature, but that is specially written to handle temperature scales.

```
>>> pm.units.temperature_scale(value, \
...     from_units=None, to_units=None,\
...     inplace=False)
```

Just like the `Conversion` instances, it respects the `unit_temperature` configuration parameter for any unspecified units. However, it makes no sense to refer to a temperature scale with any exponent other than 1, so the exponent parameter is absent.

The temperature scales and their respective differential values in kelvin are listed in Table 4.6. Note that the offsets are not included in this table.

Examples:

```
>>> pm.units.temperature(2, 'C', 'F')
3.6
>>> pm.units.temperature_scale(2, 'C', 'F')
array(35.6)
```

Table 4.6: Temperature units recognized by PYroMat

Setting	Value	Description
`K`	1 K	kelvin
`C`	1 K	degree Celsius
`R`	5/9 K	rankine
`F`	5/9 K	degree Fahrenheit
`eV`	$\approx 86.173\ 332\ 6 \times 10^{-6}$ K	electron-volt

4.6 Derived Units

Derived units are ones that are defined in terms of the fundamental units. They have no fundamental definition in nature, but are instead calculated in terms of the fundamental units.

4.6.1 Force

`Conversion` instance:	`pm.units.force`
`pyromat.config` entry:	`unit_force`
Default:	`'N'`

To alter the PYroMat unit for force quantities,

```
>>> import pyormat as pm
>>> pm.config['unit_force'] = 'kgf'
```

Force is a concept that originates with Newton's laws of motion, so it is fitting that the principle force unit be called the Newton. It is equivalent to one kg m s^{-2}, corresponding to mass times acceleration.

Force units that are derived from measures of mass (like kilogram-force and pound-force) can be calculated in terms of Newtons by calculating their weight (in Newtons) under Earth gravity. These units depend on the value assigned to g in the last call to `setup()` (see Section 4.2 of this chapter).

Table 4.7 lists the supported force units and their values in Newtons.

Table 4.7: Force units recognized by PYroMat

Setting	Value	Description
`N`	1 N	newton
`kN`	1,000 N	kilonewton
`lbf`*	≈ 4.448 221 62 N	pound-force
`lb`*	Alternate for `lbf`	
`oz`*	≈ 0.278 013 851 N	ounce

4.6.2 Energy

`Conversion` instance:	`pm.units.energy`
`pyromat.config` entry:	`unit_energy`
Default:	`'kJ'`

To alter the PYroMat unit for energy quantities,

```
>>> import pyormat as pm
>>> pm.config['unit_energy'] = 'kcal'
```

The Joule is the SI measure of energy, and it is defined as one N m, or one kg m^2 s^{-2}. The calorie is of historical importance to the scientific community, but its use has fallen out in favor of the Joule. The British thermal unit (BTU) is still broadly used in some industries (especially heating and refrigeration), but few authoritative definitions for unit systems recognize either in their contemporary standards.

The calorie has a number of alternate definitions that has made the unit somewhat problematic. One set of definitions is the energy required to raise a gram of water one degree Celsius at different "standard" conditions. Another is 1/100 of the energy required to raise a gram of water from its ice point to boiling at atmospheric pressure. In 1956, the Fifth International Conference on the Properties of Steam set the "international table calorie" to be precisely 4.1868 J. The calorie in broadest contemporary use is the thermochemical calorie, which seems to be universally understood to be precisely 4.184 J [**?**], though an original source for that definition is difficult to find. Because it is the unit adopted by the International Unions of Pure and Applied Chemistry and Physics (IUPAC and IUPAP) [**?**] PYroMat also adopts

Table 4.8: Energy units recognized by PYroMat

Setting	Value	Description
`J`	1 J	joule
`kJ`	1,000 J	kilojoule
`cal`	4.184 J	calorie
`kcal`	4,184 J	kilocalorie
`BTU`	$\approx$ 1,054.350 26 J	British thermal unit
`eV`	1.602 176 634 $\times 10^{-19}$ J	electron-volt

the thermochemical calorie.

Whatever definition is used for the calorie, the BTU may be derived from it by adjusting the quantity of water to be one pound and the temperature rise to be one degree Fahrenheit. The BTU so derived from the thermochemical calorie is approximately 1,054.350 26 J.

It is important to emphasize that it is difficult to find contemporary authorities that certify the BTU or calorie for commercial use. The ISO standard that historically defined the calorie (ISO 31-4) was withdrawn and superseded by ISO 80000-5, which makes no mention of the calorie or the BTU. NIST's special publication number 811 from which the conversion is lifted specifically lists the calorie as an "unacceptable unit."

Finally, the electron-volt is the energy required to move a single electron through a one volt potential. It is expressed in Joules as 1V $\times q$.

4.6.3 Pressure

`Conversion` instance:	`pm.units.pressure`
`pyromat.config` entry:	`unit_pressure`
Default:	`'bar'`

To alter the PYroMat unit for pressure quantities,

```
>>> import pyormat as pm
>>> pm.config['unit_pressure'] = 'Pa'
```

Pressure is a force exerted per unit area. In addition to its typical use describing fluid forces on surfaces, pressure units are also used to describe stresses and surface loads in solids. The SI unit for pressure is the pascal, which is defined as one N m^{-2} [**?**]. The bar is precisely 100,000 Pa, and is in broad use thanks to its proximity to the atmosphere. The *standard* atmosphere is internationally accepted to be precisely 101,325 Pa [**?**].

Liquid column units for pressure are defined as the increase in pressure observed beneath a column of liquid of some height. Provided sufficient measures are taken to avoid the impacts of meniscus, the liquid column pressure, p, may be calculated for a liquid with mass density, ρ, under a uniform gravitational acceleration, g, with a column height, h,

$$p = \rho g h. \tag{4.8}$$

For measures of high precision, it is obviously necessary to define a standard gravity and standard densities for the column fluids.

By default, PYroMat uses the acceleration of free fall in Earth gravity, g, to be precisely 9.8065 m s^{-2}. This value was internationally adopted in 1901 by the third General Conference on Weights and Measures (CGPM)[**?**] and is still in broad use as a "standard" value [**?**, p.45] [**?**, p.5]. However, it should be emphasized that actual weights observed vary significantly with altitude, proximity to dense geological formations, and especially latitude.

The conventional mmH_2O is calculated with precisely 1,000 kg m^{-3} water density, and the so-called 4°C mmH_2O is based on a water density of 999.972 kg m^{-3}. Actual water density in the "ambient" range 20°C to 25°C can be as low as 997 kg m^{-3}, so practical realizations of this unit in a laboratory setting are unlikely to ever be more precise than 0.3%. By default PYroMat uses the 4°C mmH_2O, but users may change this convention by altering the `dh2o` parameter in the `setup` function (see Section 4.2 of this chapter).

There are also conventional and 0°C mmHg units in broad use, but their distinctions are even less important. By default, PYroMat uses a density of 13,595.1 kg $^{-3}$ for mercury, which results in a conversion consistent with the value adopted by NIST [**?**, p.52] to the precision given.

Table 4.9: Pressure units recognized by PYroMat

Setting	Value	Description
`Pa`	1 Pa	pascal
`kPa`	1,000 Pa	kilopascal
`MPa`	1,000,000 Pa	megapascal
`bar`	100,000 Pa	bar
`atm`*	101,325 Pa	atmosphere
`Torr`*	≈ 133.322 368 Pa	Torr
`mmHg`*	≈ 133.322 387 Pa	mm mercury column
`inHg`*	≈ 3,386.388 64 Pa	inches mercury column
`mmH2O`*	≈ 9.806 375 41 Pa	mm water column
`inH2O`*	≈ 249.081 936 Pa	inches water column
`psi`	≈ 6,894.757 29 Pa	pounds per square inch
`ksi`	≈ 6,894,757.29 Pa	kips per square inch
`psf`	≈ 47.880 259 0 Pa	pounds per square foot

The "standard atmosphere" was adopted by the (CGPM) in 1954 to be precisely 101,325 newtons per square meter [**?**]. It is intended to be indicative of a global mean adjusted to sea level. This definition is in broad use both scientifically [**?**] and commercially [**?**] as value of one atm. The standard atmosphere can be adjusted using the `pstd` parameter of `setup` (see Section 4.2 of this chapter).

Because they are identical to practical precision, the Torr and the mmHg are often treated as identical units, but, there is a difference in definition. The Torr is defined so that 1 atm is precisely 760 Torr.

4.6.4 Volume

`Conversion` instance:	`pm.units.volume`
`pyromat.config` entry:	`unit_volume`
Default:	`'m3'`

To alter the PYroMat unit for volumetric quantities,

```
>>> import pyormat as pm
>>> pm.config['unit_volume'] = 'gal'
```

Volume is the measure of the size of a region in space. SI volumetric units are entirely derived from cubes of the linear units, with the liter being equivalent to 0.001 m^3.

Historically the English units for fluid ounce, pint, quart, and gallon were based on the volume occupied by specific weights of water. The English wine gallon was first implemented in 1707 as precisely 231 cubic inches, which was quite close to the gallon based on quantities of water. Though the United Kingdom abandoned it in 1824, it was adopted as the official gallon by the United States Treasury Department in 1832 [**?**]. Today, the US gallon is still defined as precisely 231 cubic inches.

The Imperial gallon, which the UK and Canada adopted in place of the wine gallon, was the volume occupied by 10 pounds of liquid water at atmospheric conditions. Today, the imperial gallon is precisely 4.546 09 liters, which is consistent to the degree that the density of water is constant.

The system of US liquid quart (1/4 gallon) and liquid pint (1/8 gallon) follow from the definition of the gallon, but it should be emphasized that there is a vast and nuanced system of specialized volumetric units that are not included in PYroMat(e.g. fluid ounce, dry quart, dry int, bushel, etc.). This decision is primarily to avoid confusion, but also to keep the focus on units of relevance to the engineering and scientific community.

Table 4.10 shows the volumetric units recognized by PYroMat.

Table 4.10: Volumetric units recognized by PYroMat

Setting	Value	Description
`cum`	1 cum	cubic meter
`m3`	Alternate for `cum`	
`cc`	$\times 10^{-6}$ cum	cubic centimeter
`cm3`	Alternate for `cc`	
`cumm`	1×10^{-9} cum	cubic millimeter
`mm3`	Alternate for `cumm`	
`L`	.001 cum	liter
`mL`	1×10^{-6} cum	milliliter
`uL`	1×10^{-9} cum	microliter
`cuin`	$0.163\ 870\ 64 \times 10^{-5}$ cum	cubic inch
`in3`	Alternate for `cuin`	
`cuft`	0.028 316 846 591 cum	cubic foot
`ft3`	Alternate for `cuft`	
`USgal`	0.003 785 411 783 cum	US gallon
`gal`	Alternate for `USgal`	
`qt`	$0.946\ 352\ 946 \times 10^{-3}$ cum	liquid quart
`pt`	$0.473\ 176\ 473 \times 10^{-3}$ cum	liquid pint
`UKgal`	0.004 546 09 cum	imperial gallon

Chapter 5

The PYroMat load process

5.1 The class registry module, `reg`

5.2 The data module, `dat`

5.3 Tools

Chapter 6

Ideal Gases

The ideal gas is one in which bombarding molecules do not exert significant forces on each other except in collisions. Under these conditions, the distance between molecules (the gas's density) is unimportant for determining thermodynamic properties. Only the gas's temperature (the speed of the molecules) is important. It is worth emphasizing that transport properties (like conductivity and diffusivity) are still impacted by density.

There are two classes in PYroMat that implement the two most widely used models: the `ig` class manages the Shomate equation of state, and `ig2` manages the so-called NASA polynomials equation of state. In either case, constant-pressure specific heat, c_p, is constructed purely as a function of temperature. Here, we re-develop the thermodynamic properties from first principles to demonstrate how c_p is sufficient to calculate them.

6.1 Properties of ideal gases

6.1.1 Ideal gas law

When they are spread so sparsely that forces between molecules are small, ideal gases are well described by the relations

$$p = nkT \tag{6.1a}$$
$$p = \rho RT \tag{6.1b}$$
$$p = \overline{\rho} R_u T. \tag{6.1c}$$

Here, p and T are the pressure and temperature of the gas. The densities are expressed in number density, n, mass density, ρ, and molar density, $\overline{\rho}$. It is clear, then, that the Boltzmann constant, k, and the ideal gas constants, R and R_u, are related,

$$kN_a = RW = R_u. \tag{6.2}$$

The Boltzmann constant is integral to the fundamental definition of the units of temperature (see Section 4.5.6), so this relationship makes it possible to calculate the universal gas constant in molar units and the ideal gas constant in mass units.

This law is empirical in its origins. It was originally formulated as an amalgamation of the independent laws of Gay-Lussac, Charles, and Boyle. The kinetic theory of gases eventually provided an independent formulation based entirely in Newton's laws. Given its importance to the study of matter and its deeply intuitive nature, it is surprising that virtually no introductory text on thermodynamic gives the subject any treatment. It is worth a brief summary here.

First, let us take that a gas is comprised of molecules that translate freely in space. They may be imagined to follow straight paths of constant velocity unless they collide with a containing wall or another molecule.

Pressure, then, is due to a near continuous stream of impacts on a surface. It only has meaning when the density of molecules is sufficiently high that their individual impacts are imperceptible. Pressure, then, is a kind of average force determined by a series of may individual

random impacts. It is possible to quantify its magnitude by describing the individual imapcts through Newtonian mechanics. If the forces due to a series of impacts in time is $F(t)$, a surface with area, A, will experience a pressure, p, based on the rate of increase of total impulse,

$$p \equiv \frac{1}{At} \int_0^t F(\tau) \mathrm{d}\tau. \tag{6.3}$$

This definition should have a well defined value in the limit where t is larger than period between individual impacts.

In a Cartesian coordinate system with z normal to a surface and positive *into* the surface, the velocity of any single molecule will have three components, $\vec{u}_1 = u_x\hat{i} + u_y\hat{j} + u_z\hat{k}$. After an elastic collision with the surface, the velocity will be $\vec{u}_2 = u_x\hat{i} + u_y\hat{j} - u_z\hat{k}$.

The impulse imparted to the surface by one such collision will be

$$\int_0^t F \mathrm{d}\tau = 2mu_z, \tag{6.4}$$

when m is the mass of the molecule. When collisions occur due to many identical molecules, their impulses accumulate

$$\begin{aligned} \int_0^t F \mathrm{d}\tau &= 2m \left(u_{z,1} + u_{z,2} + u_{z,3} + \ldots \right) \\ &= 2m \sum_i u_{z,i} \end{aligned} \tag{6.5}$$

Note that for a molecule to collide with the surface, its z-component velocity before the collision, u_z, must be positive. This results in a purely positive force (into the surface). Ideal gas molecules experiencing elastic collisions have no mechanism to pull on the surface.

The next step to predict the force on the surface requires a prediction for the rate at which collisions occur. Because faster moving molecules will travel more distance in the time interval, their collisions will be more numerous. Let the number density of molecules with z-component velocity between u_z and $u_z + \mathrm{d}u_z$ be $n'(u_z)\mathrm{d}u_z$. Here, n' is a population density function for a population of molecules based on one component of their velocity. So, n' has units number per volume

per velocity, and its integral over all velocities is precisely n, the total number density of molecules.

The number of molecules with a certain velocity that will strike an area, A, in time t will be determined by the size of the volume that can be traversed by molecules traveling at that velocity. Molecules traveling with z-component velocity u_z will traverse a length $u_z t$ in the time interval. So, the total volume occupied by molecules with that velocity that will strike the surface is Atu_z. The total number of collisions is $Atu_z n'(u_z)\mathrm{d}u_z$. So, the impulse becomes

$$\int_0^\tau F\mathrm{d}\tau = 2m\int_0^\infty Atu_z{}^2 n'(u_z)\mathrm{d}u_z \quad = mAt\int_{-\infty}^\infty u_z{}^2 n'(u_z)\mathrm{d}u_z$$

Note that only half of the gas's population will have a velocity component in the positive direction, and the other half will not cause pressure. Therefore, the 2 coefficient is canceled when the bounds of the integral are extended to include all molecules.

The integral is merely a calculation of the average value of $u_z{}^2$ over the population of molecules, and may be simplified to $n\langle u_z{}^2\rangle$ when n is simply the total number density of the molecules and $\langle u_z{}^2\rangle$ is the mean square of z-component velocity. Finally, we have obtained an expression for pressure in terms of the outward velocity component,

$$p = mn\langle u_z{}^2\rangle. \tag{6.6}$$

The last simplification to this relationship comes when we assert that gas velocity statistics are isotropic; velocity statistics are the same in all directions and do not depend on the coordinate system. That implies that $\langle u_x{}^2\rangle = \langle u_y{}^2\rangle = \langle u_z{}^2\rangle$, so

$$\langle \vec{u}^2\rangle = \langle u_x{}^2 + u_y{}^2 + u_z{}^2\rangle = 3\langle u_z{}^2\rangle. \tag{6.7}$$

Therefore, pressure is proportional to the mean square of molecular translational velocity,

$$p = 3mn\langle u^2\rangle. \tag{6.8}$$

When this relationship is substituted into the ideal gas law above, it provides a purely mechanical interpretation for temperature as well,

$$\langle \frac{1}{2}mu^2\rangle = \frac{3}{2}kT. \tag{6.9}$$

Temperature is a measure of the average *thermal* energy of the gas. Higher temperature means higher kinetic energy, and the Boltzmann constant relates the two.

This development was quick and it neglects to address mixtures of molecules of different masses. However, the same approach may be used quite intuitively to show that Dalton's law for partial pressures also follows from these basic assumptions.

6.1.2 Internal energy

What happens to heat and work as they are added to an ideal gas depends on the structure of the gas molecule. If there are chemical, atomic, or phase changes, the species can be modeled as vanishing and being replaced by a new substance with its own properties. The ideal property models are, therefore, concerned with how energy is stored in a molecule that is neither changing its fundamental structure nor changing phase. This remaining energy will be exhibited entirely as the thermal energy in (**??**) and internal vibration, rotation, or electrical motions inside the molecule, for which we have not yet accounted.

Perfect gases are ideal gases, but not all ideal gases are perfect. The molecules of an ideal gas collide elastically and have no further interactions with their surroundings. However, the molecules of a perfect gas are further assumed to neither spin nor vibrate. As a result, any molecule more complicated than a single atom does not form a perfect gas.

When a gas composed of a monoatomic molecule like argon is heated (without atomic, chemical, or phase changes) the energy can only be stored in the thermal translational energy described in (6.9). A sample of N molecules of such a gas would have thermal energy $N\frac{3}{2}kT$, so the energy per mole is

$$e - e_0 = \frac{N_a}{W}\frac{3}{2}kT = \frac{3}{2}RT \tag{6.10}$$

when e_0 is the internal energy due to the atomic, chemical or phase changes we have not yet considered. Note that when this is expressed in molar units, R becomes R_u, and when it is expressed in molecules, R is replaced by the Boltzmann constant, k.

Non-perfect ideal gases are made of more complicated molecules can store energy in more complicated ways; they vibrate, they rotate (spin), and they have means of storing energy electrically. They do this because the can; they have more degrees of freedom.

The 3 in (6.10) first appeared in (6.7) because all ideal gases are free to translate in three directions. These three thermal degrees of freedom are the only ones that contribute to measurements of temperature, but complex molecules have more degrees of freedom. If these are included in the internal energy as well, a molecule that is free to vibrate, rotate, and translate in f degrees of freedom will have an internal energy

$$e - e_0 = \frac{N_a}{W}\frac{f}{2}kT = \frac{f}{2}RT. \tag{6.11}$$

The intuitive assumption might be that f is constant and a property of each molecule. This assumption implicitly asserts that all modes of vibration, rotation, and translation have an equal fraction of the total energy so it is called the *equipartition* assumption. It results in an elegant formulation, but it completely fails to predict the actual behavior of molecules in gases.

Instead, it is necessary to stipulate that f is a statistical quantity that can change with the state. It can be thought of as the number of "active" degrees of freedom of each molecule. Its minimum is 3, but it can increase to a maximum that will depend on the complexity of the molecule and the energy of the collisions it endures. For example, at very low temperatures, molecules may not vibrate much; instead they may just bang around and spin like rigid bodies (see Figure 6.1 below). However, at higher temperatures, there may be enough energy to excite vibrations inside the molecules.

Fortunately, since all ideal gases (perfect or not) are presumed to collide elastically, the equilibrium distribution of energy should depend on neither the rate of collisions nor the distance separating the molecules. Obviously, this assumption will break when molecules are packed in closely with one another. However, so long is that is true, the active degrees of freedom and the internal energy will both only be a function of temperature (the average kinetic energy).

$$e(T) - e_0 = \frac{f(T)}{2}RT \tag{6.12}$$

Better insights into the behavior of f will be seen when we examine specific heats below.

6.1.3 Enthalpy

As established above, we will construct all of the properties of the ideal gas in terms of its specific heat, but in order to derive reliable relations for the specific heats, it is important to first re-examine enthalpy.

Recall from Section 1.1.7 that the definition for enthalpy is $e + pv$. For an ideal gas, pv may be substituted for RT, so an ideal gas will have enthalpy

$$h(T) = e(T) + RT. \tag{6.13}$$

For all ideal gases, enthalpy and internal energy are only functions of temperature, and they are related by RT.

6.1.4 Specific heats

Recall from Section 1.1.9 that the constant-volume and constant-pressure specific heats are

$$c_v = \left(\frac{\partial e}{\partial T}\right)_v$$
$$c_p = \left(\frac{\partial h}{\partial T}\right)_p$$

Because both internal energy and enthalpy are only functions of temperature, this relationship is relatively simple,

$$c_v = \frac{f(T)}{2} R \left(1 + T\frac{f'(T)}{f(T)}\right) \approx \frac{f(T)}{2} R \tag{6.14}$$
$$c_p = c_v + R \tag{6.15}$$

The approximation in (6.14) only holds when $f(T)$ is very nearly constant.

Figure 6.1 shows the quantities $2c_v/R$ for six gases with increasing molecular complexities. The perfect gases, He and Ar, behave

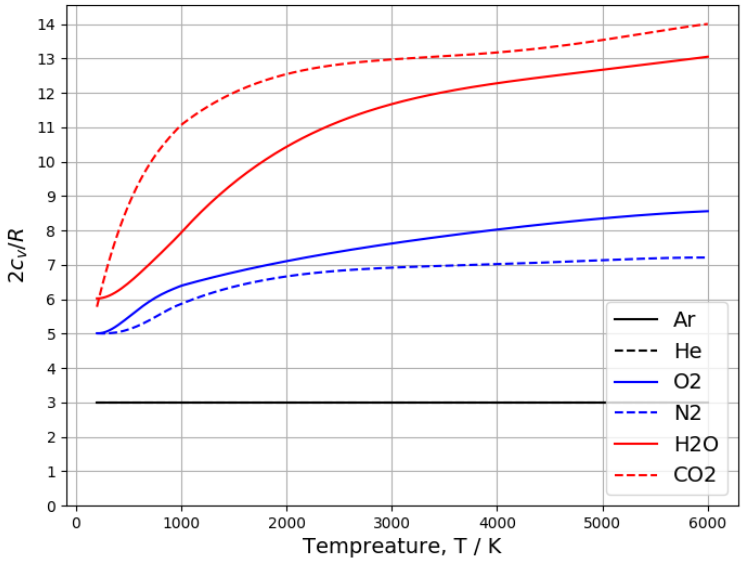

Figure 6.1: Approximate mean active degrees of freedom in selected ideal gas models

precisely as predicted with exactly three degrees of freedom. All four polyatomic molecules start with quasi-rigid motion at low temperatures before adopting higher degrees of freedom at high temperatures.

For H_2O, rigid motion means six degrees of freedom: three coordinates of translation and three axes of rotation. For diatomic molecules like O_2 and N_2, however, their symmetry robs them of one of their axes of rotation, so they only exhibit five degrees of freedom at low temperatures. Since CO_2 is also an axisymmetric molecule, it can be seen asymptotically approaching 5 at low temperatures as well.

6.1.5 Entropy and enthalpy revisited

Recall from Section 1.1.8 that the entropy of any substance changes like

$$T\mathrm{d}s = \mathrm{d}h - v\mathrm{d}p.$$

For the ideal gas, this can be simplified by substituting RT/p for v and integrating to obtain

$$s - s_0 = \int c_p(T)\frac{\mathrm{d}T}{T} - R\ln\left(\frac{p}{p^\circ}\right). \tag{6.16}$$

when p° is a reference pressure. For a perfect gas,

$$s - s_0 = c_p\ln\left(\frac{T}{T_0}\right) - R\ln\left(\frac{p}{p^\circ}\right). \tag{6.17}$$

when T_0 is a reference temperature.

Evaluating the pressure portion of (6.17) is straightforward, but it will be seen below that the temperature dependence is more nuanced. For that reason, it is often treated alone,

$$s^\circ(T) = s_0 + \int c_p(T)\frac{\mathrm{d}T}{T}, \tag{6.18}$$

where s° is the entropy at the reference pressure, p°.

An identical approach can be taken for enthalpy, but with an even simpler result.

$$h - h_0 = \int c_p(T)\mathrm{d}T \tag{6.19}$$

For a perfect gas,

$$h - h_0 = c_p \left(T - T_0 \right). \tag{6.20}$$

In the NIST-JANAF tables, the specific heat is calculated theoretically from the molecular and atomic structure of each atom, but specific heat is also readily validated by calorimetry. Entropy's integration constant, s_0, is chosen so the values agree with absolute calculations for the species entropy from Boltzmann's statistical model for entropy. Since no such model exists for enthalpy, h_0 is determined by a separate approach described in Section 6.1.7.

The models in the `ig` and `ig2` classes use systems of coefficients to form piece-wise polynomials for $c_p(T)$ instead of using the detailed models recorded in the NIST-JANAF tables. These are ideal for computational codes because they give good numerical performance, but they are not valid down to the low temperatures included by the original tables.

6.1.6 Other properties

Once $c_p(T)$ is well defined, it is also possible to evaluate internal energy,

$$e(T) = h(T) - RT, \tag{6.21}$$

constant-volume specific heat,

$$c_v(T) = c_p(T) - R, \tag{6.22}$$

specific heat ratio,

$$\gamma(T) = \frac{c_p(T)}{c_p(T) - R}, \tag{6.23}$$

speed of sound

$$a(T) = \sqrt{\gamma(T) R T}, \tag{6.24}$$

and others.

6.1.7 Enthalpy of formation

When a substance is formed either by nuclear, chemical or phase change, energy is nearly always released or consumed. This energy is accounted for by a property called the *enthalpy of formation*. It is the energy required to form a substance, so exothermic reactions have negative enthalpies of formation. The ideal gas data on which PYroMat's ideal gas classes do not consider nuclear reactions; only chemical reactions and phase changes.

One might imagine a reactor with a mixture of reactants flowing in and a mixture of products flowing out. In the simplest case, we should imagine the products to be made entirely of the substance we wish to study, so the reactants will be only those that are absolutely necessary for forming it and in the correct proportions.

$$a\mathrm{A} + b\mathrm{B} + \ldots \rightarrow \mathrm{A}_a\mathrm{B}_b\ldots \tag{6.25}$$

Many such chemical reactions release or consume vast amounts of energy. Usually, this results in cooling or heating of the substance as it changes. When heat and work are neither added nor removed from the system, an energy balance mandates an isenthalic process,

$$\begin{aligned} h_{\text{reactants}} &= h_{\text{products}} \\ a\overline{h}_{\mathrm{A}} + b\overline{h}_{\mathrm{B}} + \ldots &= \overline{h}_{\mathrm{A}_a\mathrm{B}_b\ldots} \end{aligned} \tag{6.26}$$

when $\overline{h}$ is enthalpy in molar units.

This result is often counter intuitive. Since ideal gas enthalpy is only a function of temperature, it seems like an isenthalpic process should also be isothermal. However, when chemical or phase changes take place, the enthalpy curves of the products and reactants can be dramatically different.

Fig. 6.2 shows enthalpy curves for a hypothetical set of reactants and products. Not only may the two have dissimilar slopes, but the curves may have large offsets separating them. When the process is isenthalpic, these offsets cause temperature changes shown in red. In the case shown, the reactants have higher enthalpy than the products, so the additional energy is absorbed thermally by the products, causing an increase in temperature. On a molecular level, the effect is like

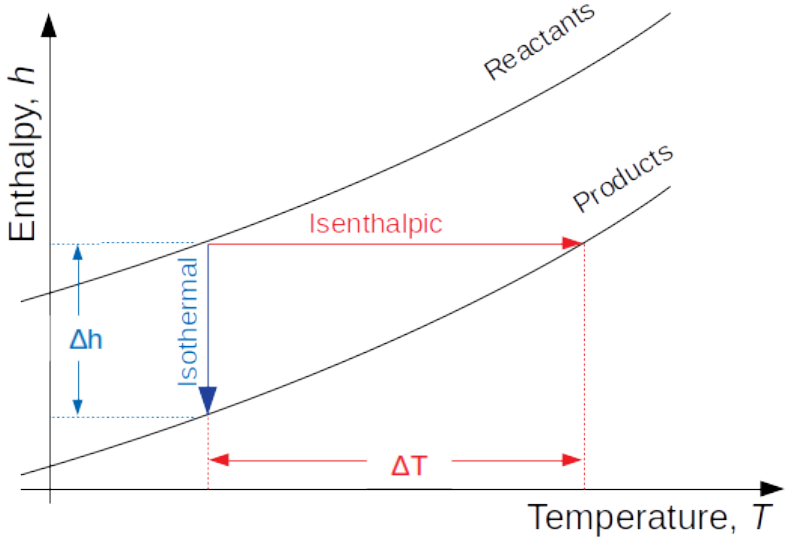

Figure 6.2: Isenthalpic and isothermal reactions on an h-T diagram.

allowing two magnetic marbles to roll near one another on a flat table. Even if neither has much velocity to begin with, after they collide, they will be sent off quickly spinning and rolling. The same happens in an exothermic reaction.

If the hypothetical reactor were modified to add or extract heat so that the temperature of the reaction were constant, the process would adopt the vertical blue line instead. In an isothermal reaction, the amount of thermal energy exhibited by the substance is constant, so any release or consumption of heat will have come from a chemical reaction (including a change in the substance's active degrees of freedom, f). Therefore, the change in enthalpy between the two curves is the enthalpy consumed by the chemical reaction reaction.

In the diagram, the enthalpy is seen decreasing in order to maintain the temperature, so heat was removed, and the reaction is called exothermic. For such a reaction, the energy balance would be

$$a\overline{h}_{\mathrm{A}} + b\overline{h}_{\mathrm{B}} + \ldots + \Delta h = \overline{h}_{\mathrm{A}_a\mathrm{B}_b\ldots} \tag{6.27}$$

Here, the change in enthalpy, Δh, is positive when energy is added to maintain a constant temperature throughout the reaction, so in the exothermic reaction depicted in fig. 6.2, Δh would be negative. In general Δh is also a function of temperature because the properties of the two gas mixtures are not parallel.

The value of $\Delta h(T)$ also depends on which reactants are selected to form the product. For example, one might select atomic oxygen and atomic hydrogen to form water, but it would be just as valid to chose the more common diatomic hydrogen and oxygen. These two reactions would exhibit different Δh values because the energies of the starting substances are different. If one wishes to establish a value for Δh that is clearly defined and indicates the energy required to form the substance, it is necessary to specify a standard set of "reference" substances, relative to which all other substances are derived.

Every element is given exactly one **reference substance**, which may include phase changes (like in the case of the metals). The enthalpy of formation for the reference substance is arbitrarily declared to be zero at all temperatures, since it is imagined to be a fundamental building block from which compounds are built.

The *enthalpy of formation*, $\Delta_f h$, is the energy consumed by the reaction when all of the reactants are reference substances in the proper proportions to produce the product.

Many of the reference substances are atomic gases or pure liquids and crystals (like argon and aluminum). Others are selected to be diatomic gases because of their abundance (like hydrogen, oxygen, and nitrogen).

6.1.8 Properties of mixtures

The discussion so far has applied only to properties of pure ideal gases. The composition of a gas mixture is conventionally defined in either mass or molar quantities of the constituent pure gases. In this section, we establish methods by which properties can be calculated from the properties of the components.

In extensive units, a volume, V, might contain many individual gas species, each with total mass, m_i, or total mole count, N_i. They are

related by the pure gas's molecular weight,

$$m_i = W_i N_i. \tag{6.28}$$

Mass and mole fractions are the fractions of a gas composed of a single pure substance. The total mass and count of gas in the volume is merely the sum of all the constituents, $m = \sum m_i$, and $N = \sum N_i$. These let us more conveniently express the composition in mass and mole fractions, respectively,

$$y_i \equiv \frac{m_i}{\sum m_k} \tag{6.29a}$$

$$\chi_i = \equiv \frac{N_i}{\sum N_k}. \tag{6.29b}$$

Note that the fractional quantities, y_i and χ_i, are not dependent on the choice of units for mass or mole count. Observe that, by definition, $\sum y_i = \sum \chi_i = 1$.

Mixture density is the total mass or mole count of all species divided by the volume they occupy.

$$\rho \equiv \frac{\sum m_i}{V} \tag{6.30a}$$

$$\overline{\rho} \equiv \frac{\sum N_i}{V} \tag{6.30b}$$

If ρ_i and $\overline{\rho}_i$ were the densities of constituent i at the same temperature and pressure as the total mixture, then it is important to emphasize that m_i/V and N_i/V are *not* ρ_i and $\overline{\rho}_i$. That question is addressed below.

Molecular weight of a gas mixture has the same definition as the molecular weight of a pure gas; it is the mass per mole count of total gas. It can be calculated from the constituent molecular weights using

either mole fractions or mass fractions,

$$W \equiv \frac{\sum m_i}{\sum N_j} \tag{6.31a}$$

$$= \frac{\sum N_i W_i}{\sum N_j}$$

$$= \sum \chi_i W_i \tag{6.31b}$$

$$= \frac{\sum m_i}{\sum m_j / W_j}$$

$$= \left(\sum \frac{y_j}{W_j} \right)^{-1}. \tag{6.31c}$$

Partial pressure, p_i, is the pressure force exerted on a surface due only to collisions of a single constituent, i. Section 6.1.1 will help understand what is meant by this idea. It is the pressure force that would be measured if all other constituent gases were removed.

$$p_i \equiv \frac{N_i}{V} R_u T$$

$$= \chi_i \frac{N}{V} R_u T$$

$$= \chi_i \overline{\rho} R_u T \tag{6.32}$$

The same pressure may be calculated from mass fraction,

$$p_i = \frac{N_i}{V} R_u T$$

$$= \frac{m_i}{W_i V} \rho R_u T$$

$$= y_i \rho R_i T \tag{6.33}$$

The ideal gas constant, R_i, is the mass-based gas constant for only that constituent, $R_i = R_u / W_i$.

Total pressure, p, is the pressure force actually experienced by a surface due to all of the constituent gases. By definition,

$$p = \sum p_i$$

$$= \overline{\rho} R_u T \tag{6.34}$$

Observe, also, that

$$\chi_i = \frac{p_i}{p}. \tag{6.35}$$

The total pressure can also be calculated in terms of mass density. Using (6.31c),

$$\begin{aligned} p &= \sum p_i \\ &= \sum y_i \rho R_i T \\ &= \sum \frac{y_i}{W_i} \rho R_u T \\ &= \rho R T. \end{aligned} \tag{6.36}$$

The total mixture gas constant appears naturally.

The mixture gas constant, R, appears naturally when calculating total pressure from total mass density, and it is calculated in precisely the same way as a pure gas constant, but the mixture molecular weight is used instead,

$$R \equiv \frac{R_u}{W} \tag{6.37a}$$

$$= \frac{R_u}{\sum \chi_i W_i}$$

$$= \left(\frac{\chi_i}{R_i} \right)^{-1} \tag{6.37b}$$

$$= R_u \sum \frac{y_i}{W_i}$$

$$= \sum y_i R_i \tag{6.37c}$$

Densities can be calculated from the same properties of the com-

ponent species using the ideal gas law. For mass density,

$$\begin{aligned}
\rho &= \frac{p}{RT} \\
&= \frac{p}{\sum y_i R_i T} \\
&= \left(\sum \frac{y_i}{\rho_i(T,p)} \right)^{-1} && \text{(6.38a)} \\
&= \frac{p}{\left(\sum \chi_i / R_i\right)^{-1} T} \\
&= \sum \chi_i \rho_i(T,p) && \text{(6.38b)}
\end{aligned}$$

For molar density, no such complexity is needed, since the universal gas constant is the same for all gases,

$$\overline{\rho} = \frac{p}{R_u T}$$

Internal energy, enthalpy, entropy, and other bulk properties are merely the sum of all the values contributed by each of the constituent gases. By definition in an ideal gas, the presence of other gases do not affect the behavior of each of the pure constituents. Therefore, a bulk property may be calculated from the properties of the pure substances under equivalent temperature and pressure,

$$\begin{aligned}
\left(\sum m_j \right) \phi(T,p) &= \sum m_i \phi_i(T,p) \\
\left(\sum N_j \right) \overline{\phi}(T,p) &= \sum N_i \overline{\phi}_i(T,p)
\end{aligned}$$

Therefore,

$$\begin{aligned}
\phi(T,p) &= \sum y_i \phi_i(T,p) \\
\overline{\phi}(T,p) &= \sum \chi_i(T,p) \overline{\phi}_i
\end{aligned}$$

This is hardly a proof, but for it is sufficient for the scope of this document to assert that it is true for bulk properties.

This identity may be applied to internal energy, enthalpy, entropy, and all the properties derived from them, so

$$e(T) = \sum y_i e_i(T) \tag{6.39a}$$

$$h(T) = \sum y_i h_i(T) \tag{6.39b}$$

$$s(T,p) = \sum y_i s_i(T,p) \tag{6.39c}$$

and

$$\overline{e}(T,p) = \sum \chi_i \overline{e}_i(T,p) \tag{6.40a}$$

$$\overline{h}(T,p) = \sum \chi_i \overline{h}_i(T,p) \tag{6.40b}$$

$$\overline{s}(T,p) = \sum \chi_i \overline{s}_i(T,p). \tag{6.40c}$$

6.2 The ideal gas collection

There are several classes that implement various data models for ideal gas properties. Their interfaces are all standardized so that very little difference should be apparent to the user, except that the ideal gas mixture class has some extra methods associated with its composition.

All property methods accept any two of temperature, density, or pressure to specify the state.

6.2.1 The Shomate equation: `ig`

PYroMat's `ig1` class is built on the Shomate equation for constant-pressure specific heat c_p. This is the formulation used by the NIST webbook [**?**]. Despite the wise range of substances represented, it has the advantage of using a simple standard piecewise formulation for specific heat. However, it does suffer from certain limitations.

The Shomate equation takes the form

$$\theta = \frac{T}{T_s} \tag{6.41}$$

$$c_p(t) = c_0 + c_1\theta + c_2\theta^2 + c_3\theta^3 + \frac{c_4}{\theta^2}, \tag{6.42}$$

where the scaling temperature, T_s is 1000K for all species. The decision to scale the temperature by a large value has the effect of scaling τ so that it will not be much larger than 5 or 6. That helps reduce numerical errors in high-order polynomials.

Because of its simplicity, the Shomate equations lack the degrees of freedom to express specific heat over wide ranges, so data are usually given in piece-wise formulations. For example, tungsten dioxide (WO_2), has a set of coefficients for 298K $\leq T <$ 1100K and 1100K $\leq T \leq$ 6000K.

The enthalpy can be explicitly calculated from (**??**),

$$\begin{aligned} h(T) &= h_0 + \int c_p(T)\mathrm{d}T \\ &= h_0 + T_s \int c_p(t)\mathrm{d}t \\ &= T_s \left(c_0 t + \frac{c_1}{2}t^2 + \frac{c_2}{3}t^3 + \frac{c_3}{4}t^4 - \frac{c_4}{t} + c_5 \right). \end{aligned} \tag{6.43}$$

It is important to emphasize that h_0 is not the same as the enthalpy of formation, Δh_f°. Instead, it is merely an integration constant, which can be alternately expressed as a new coefficient, c_5.

Because of the temperature term in the denominator, no multiple of T_s appears in entropy when the integration is changed to t,

$$\begin{aligned} s(T,p) &= s^\circ(T_{ref}) + \int_{T_{ref}}^{T} \frac{c_p(\tau)}{\tau}\mathrm{d}\tau - R\ln\left(\frac{p}{p_{ref}}\right) \\ &= s^\circ(T_{ref}) + \int_{T_{ref}}^{T} \frac{c_p(\tau)}{\tau}\mathrm{d}\tau - R\ln\left(\frac{p}{p_{ref}}\right) \\ s(T,p) &= c_0 \ln t + c_1 t + \frac{c_2}{2}t^2 + \frac{c_3}{3}t^3 - \frac{c_4}{2t^2} + c_6. \end{aligned} \tag{6.44}$$

Just like in the enthalpy integral, a new coefficient, c_6, has been introduced to represent the integration constant.

Internal energy is readily calculated from the definition of enthalpy in (**??**),

$$e(T) = h(T) - RT \tag{6.45}$$

Table 6.1: HPD data file elements for the `ig` class

Name	Type	Description
`id`	`str`	Unique substance identifier string
`class`	`='ig'`	Class identifier string
`doc`	`str`	Documentation string
`atoms`	`dict`	Keys are elemental atoms, and values are their integer quantities in the substance.
`mw`	`float`	The molecular weight.
`Tlim`	`list`	A sorted list of $N+1$ floating point temperatures. Middle values define the boundaries between piece-wise coefficient ranges. High and low values define the limits of the model.
`C`	nested `list`	The value, $C[i][j]$, corresponds to coefficient c_j in the temperature interval `Tlim[i]` to `Tlim[i+1]`.
`TAB`	nested `list`	Optional table of truth values published by NIST used for validation

There is a similarly simple relationship to determine constant-volume specific heat and specific heat ratio,

$$c_v(T) = c_p(T) - R \tag{6.46}$$

$$\gamma(T) = \frac{c_p(T)}{c_p(T) - R} \tag{6.47}$$

6.2.2 The NASA polynomial: `ig2`

The so-called "NASA polynomials" are a piece-wise empirical formulation to the specific heat of an ideal gas. They predate the latest formulation of the NIST-JANAF tables, and are even used for reference. Unlike the Shomate equation, there is no $1/t^2$ term, there is no attempt to scale temperature prior to evaluating the polynomial, and properties are scaled with respect to the ideal gas constant.

$$c_p(T) = R\left(c_0 + c_1 T + c_2 T^2 + c_3 T^3 + c_4 T^4\right) \tag{6.48}$$

There are nearly identical formulations for enthalpy,

$$h(T) = R\left(c_0 T + \frac{c_1}{2} T^2 + \frac{c_2}{3} T^3 + \frac{c_3}{4} T^4 + \frac{c_4}{5} T^5 + c_5\right), \tag{6.49}$$

and entropy

$$s^\circ(T) = R\left(c_0 \ln(T) + c_1 T + \frac{c_2}{2} T^2 + \frac{c_3}{3} T^3 + \frac{c_4}{4} T^4 + c_6\right). \tag{6.50}$$

Here, just as in the Shomate equations, c_5 and c_6 are introduced as integration constants in enthalpy and entropy.

Internal energy is readily calculated from the definition of enthalpy in (**??**),

$$e(T) = h(T) - RT \tag{6.51}$$

There is a similarly simple relationship to determine constant-volume specific heat and specific heat ratio,

$$c_v(T) = c_p(T) - R \tag{6.52}$$

$$\gamma(T) = \frac{c_p(T)}{c_p(T) - R} \tag{6.53}$$

There is some overlap in the species offered by the two classes. A mild preference has been given to the older NASA polynomials for two reasons:

- Most of the NASA models have wider ranges of validity than the Shomate models.
- Some of the Shomate data have been found to suffer from discontinuity errors at the piecewise boundaries.

Otherwise, the two have been found to be sufficiently equivalent that most users will not find a reason to wonder.

Table 6.2: HPD data file elements for the `ig2` class

Name	Type	Description
`id`	`str`	Unique substance identifier string
`class`	`='ig'`	Class identifier string
`doc`	`str`	Documentation string
`atoms`	`dict`	Keys are elemental atoms, and values are their integer quantities in the substance.
`pref`	`float`	Reference pressure in Pascals
`mw`	`float`	The molecular weight.
`Tlim`	`list`	A sorted list of $N + 1$ floating point temperatures: Middle values define the boundaries between piece-wise coefficient ranges. High and low values define the limits of the model.
`C`	nested `list`	The value, $C[i][j]$, corresponds to coefficient c_j in the temperature interval `Tlim[i]` to `Tlim[i+1]`.

6.2.3 The ideal gas mixture: `igmix`

The PYroMat ideal gas mixture class defines methods to calculate properties of a static mixture of ideal gases. For improved computational efficiency, properties like the mixture molecular weight, mole fractions, and mass fractions, are calculated ahead of time and stored in class instances for later use.

Section **??** already shows how properties of a mixture can be calculated from the properties of the constituent gases. The implementation of all but entropy is quite straightforward. In all of the ideal gas codes, the entropy at the reference pressure, s°, is calculated in a separate efficient internal method. For that reason, the `igmix` class calls on these methods directly and then treats the pressure dependence separately.

When calculating in molar units,

$$\overline{s}(T,p) = \sum \chi_i \; \overline{s}_i^\circ(T) - \sum \chi_i R_u \ln\left(\frac{p}{p_i^\circ}\right)$$

Note that in this formulation, the constituents are not required to have the same reference pressure. This can be dealt with by calculating an effective mixture reference pressure.

$$\begin{aligned}\sum \chi_i \ln\left(\frac{p}{p_i^\circ}\right) &= \sum \chi_i \left(\ln p - \ln p_i^\circ\right) \\ &= \ln p - \sum \chi_i \ln p_i^\circ \\ &= \ln\left(\frac{p}{p_{mix}^\circ}\right)\end{aligned}$$

when

$$p_{mix}^\circ = \exp\left(\sum \chi_i \ln p_i^\circ\right). \tag{6.54}$$

This is a log-weighted average of the constituent reference pressures. In the event that they are all the same, the mixture reference pressure is also the same.

It can be shown that a comparable result is achieved in mass units,

Table 6.3: HPD data file elements for the `igmix` class

Name	Type	Description
`id`	`str`	Unique substance identifier string
`class`	`='igmix'`	Class identifier string
`doc`	`str`	Documentation string
`bymass`	`bool`	Indicates whether contents are listed by mass or by mole count
`contents`	`dict`	Specifies the mixture composition: Keys are pure ideal gas id strings, values are quantities. The quantities will be normalized after load, so they do not need to add to one.

so

$$s(T,p) = \sum \chi_i \overline{s}_i^\circ(T) - R \ln\left(\frac{p}{p_{mix}^\circ}\right) \tag{6.55a}$$

$$\overline{s}(T,p) = \sum \chi_i \overline{s}_i^\circ(T) - R_u \ln\left(\frac{p}{p_{mix}^\circ}\right) \tag{6.55b}$$