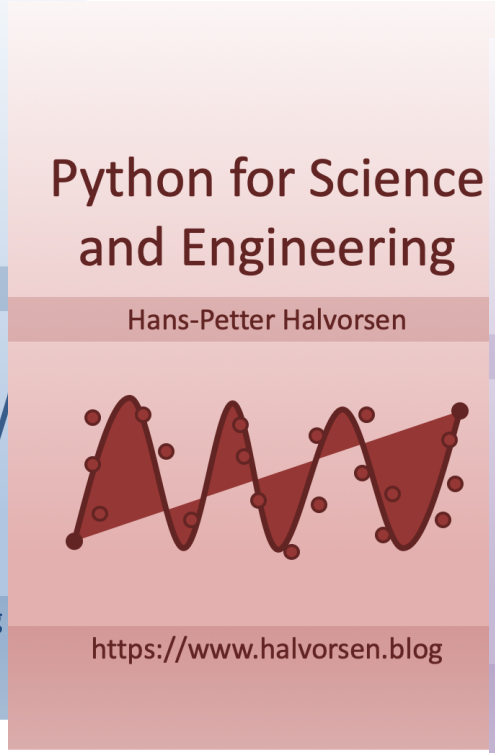
Numerical Integration in Python

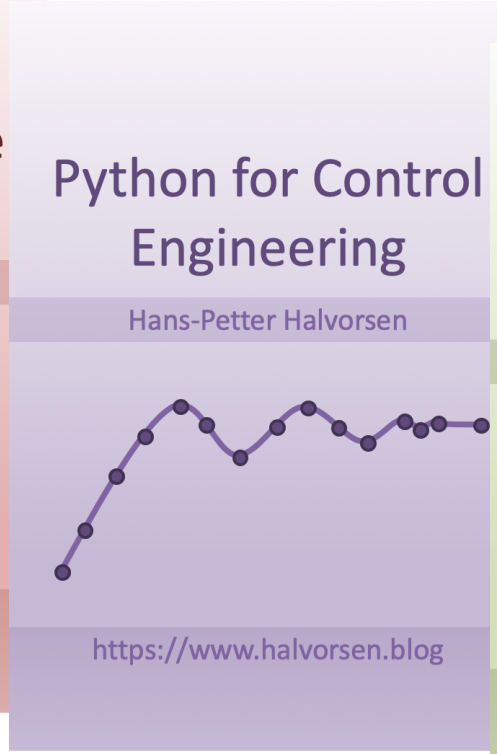
Hans-Petter Halvorsen

Free Textbook with lots of Practical Examples

https://www.halvorsen.blog/documents/programming/python/

Additional Python Resources





https://www.halvorsen.blog/documents/programming/python/

Contents

- Integrals
- Numerical Integration
- Python Examples

It is assumed that already know the basics about integrals from mathematics courses and that you want to use Python to find numerical solutions

Integrals

The Indefinite Integral

The indefinite integral of f(x) is a FUNCTION $F(x)$

$$f(x) \Longrightarrow \int f(x) = F(x) + c$$

$\int$ is called the Integral symbol

Where $\frac{dF(x)}{dc} = f(x)$

c is a constant

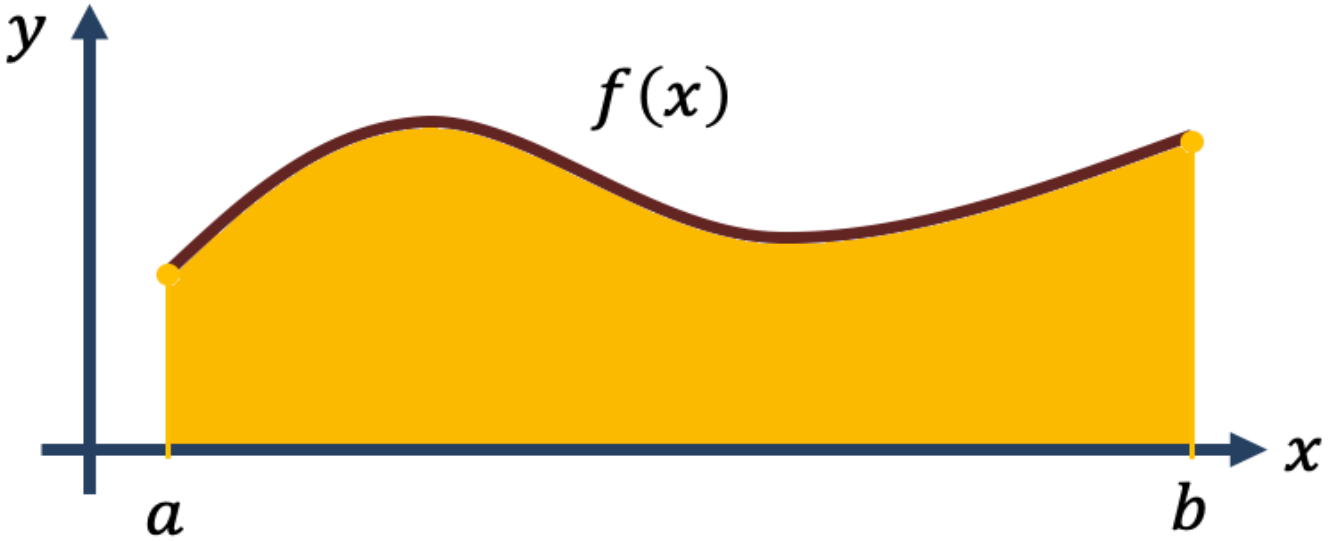
The Definite Integral

The definite integral of f(x) is a NUMBER and represents the area under the curve f(x) from $x = a$ to $x = b$.

$$f(x) \Longrightarrow \int_a^b f(x)$$

Since the topic is Numerical Integration in Python, we will focus on the **Definite Integral**

Integration



$\int_a^b f(x)dx$ = Area under the curve (Yellow color)

Example

Given the function:

$$f(x) = x^2$$

We know that the exact solution is:

$$\int_0^a x^2 dx = \frac{a^3}{3}$$

The integral from 0 to 1 is:

$$\int_0^1 x^2 dx = \frac{1}{3} \approx 0.3333$$

We will take it step by step to see how we can solve this using Python

Example

$$f(x) = x^2$$

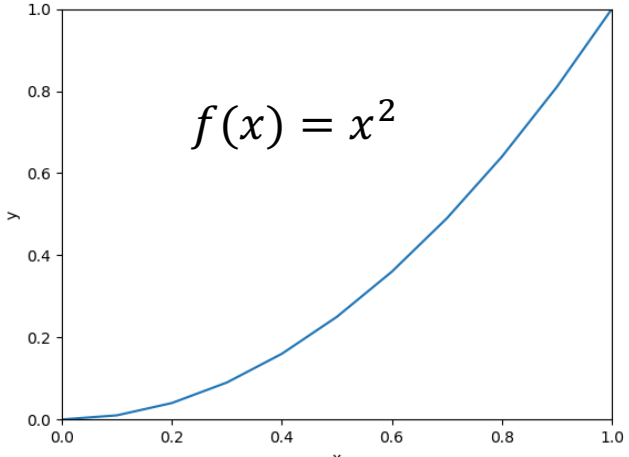
$$\int_0^1 x^2 dx = \frac{1}{3} \approx 0.3333$$

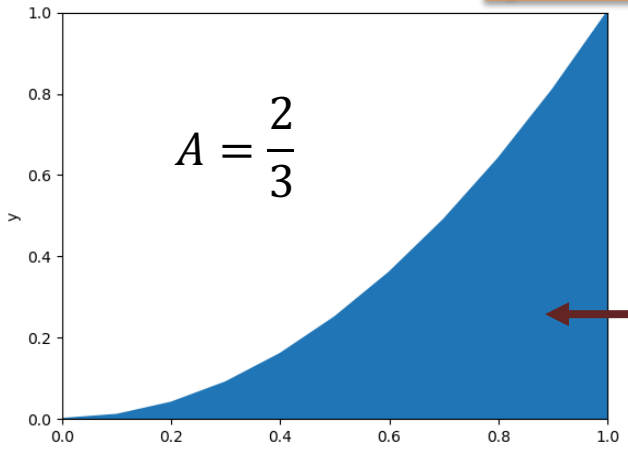
```
import matplotlib.pyplot as plt
import numpy as np

xstart = 0
xstop = 1.1
increment = 0.1

x = np.arange(xstart,xstop,increment)
y = x**2

plt.plot(x, y)
plt.xlabel('x')
plt.ylabel('y')
plt.axis([0, 1, 0, 1])
plt.fill_between(x,y)
plt.show()
```





$$A = \frac{1}{3} \approx 0.33$$

Numerical Integration Approach

Figure 1

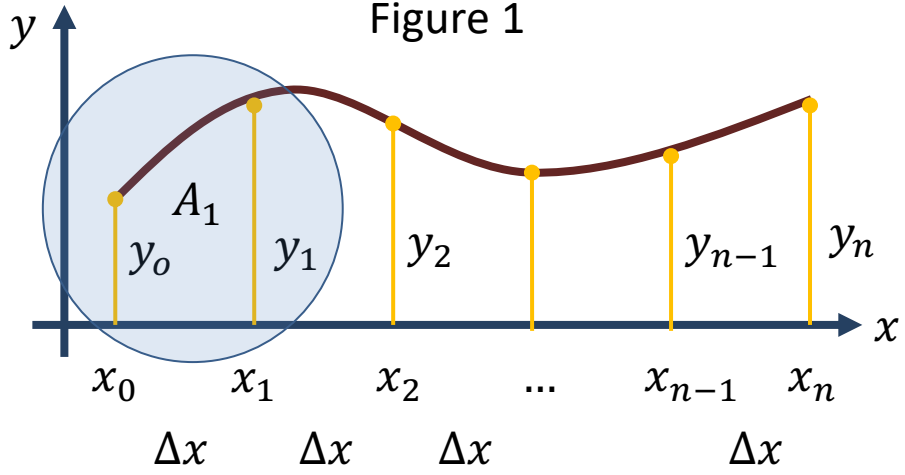
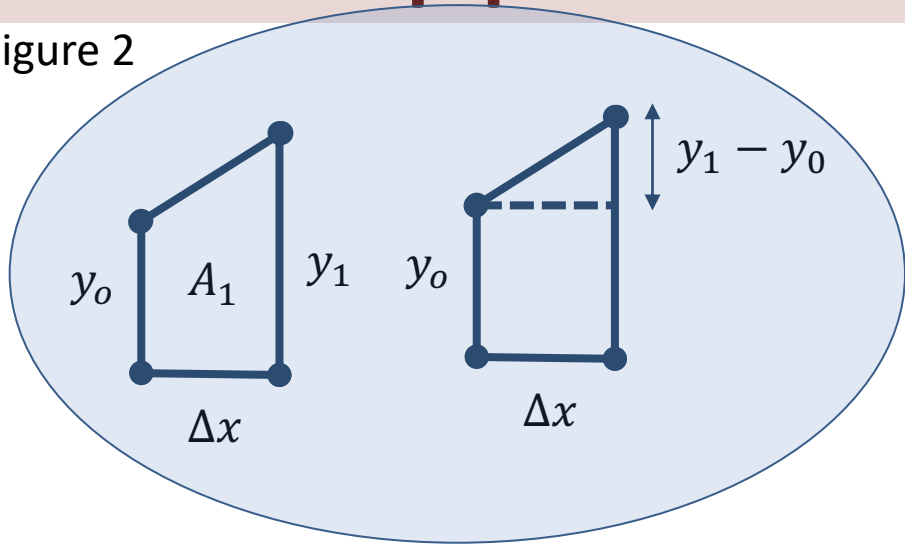


Figure 2



From Figure 2: $A_1 = y_0\Delta x + \frac{1}{2}(y_1 - y_0)\Delta x$

This gives:

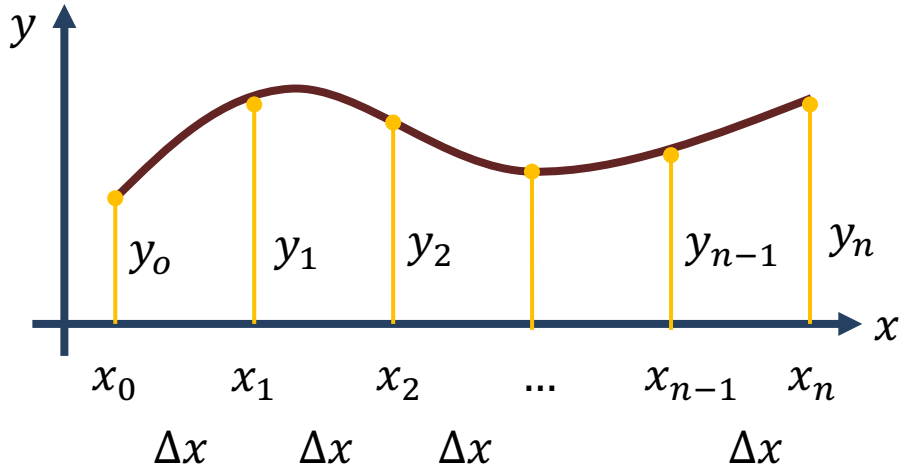
$$A_1 = y_0\Delta x + \frac{1}{2}y_1\Delta x - \frac{1}{2}y_0\Delta x = \frac{1}{2}y_0\Delta x + \frac{1}{2}y_1\Delta x$$

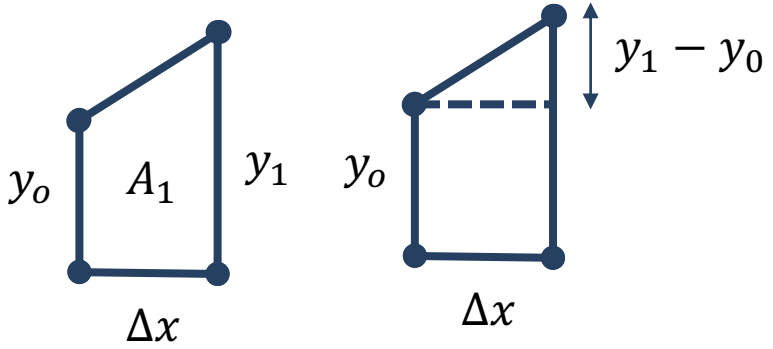
Finally:

$$A_1 = \frac{(y_0+y_1)\Delta x}{2}$$

Total Area: $A = A_1 + A_2 \ldots + A_n$

Numerical Integration





$$A_1 = \frac{(y_0 + y_1)\Delta x}{2}$$

$$A = A_1 + A_2 \ldots + A_n$$

This gives the total Area under the curve:

$$A = \sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)/2$$

Typically Δx is constant, which gives:

$$A = \frac{\Delta x}{2} \sum_{i=1}^{n-1} y_{i+1} + y_i$$

Numerical Integration

- Given y = f(x) the approximation of the Area (A) under the curve can be found dividing the area up into rectangles and then summing the contribution from all the rectangles
- This is known as the **Trapezoid** rule.

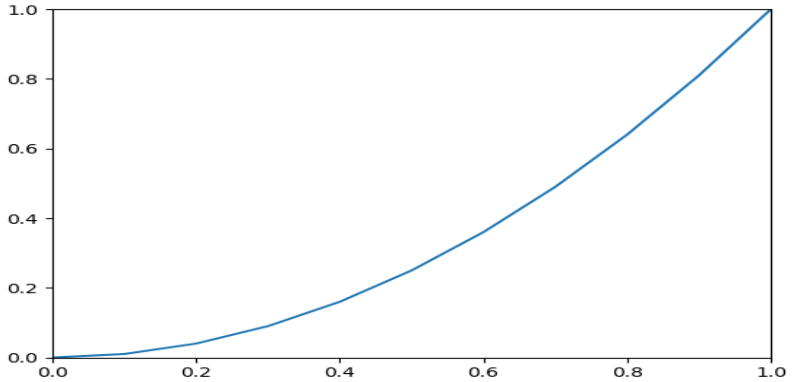
$$\int_a^b f(x)dx \quad \Rightarrow \quad A = \frac{\Delta x}{2} \sum_{i=1}^{n-1} y_{i+1} + y_i$$

We will implement and use this rule in Python, both from scratch and using the SciPy library

Examples

Python Code

$$f(x) = x^2$$



Results from the Python code:

```
A = 0.3350000000000001
```

Which is a good approximation when we now the exact answer is A = 1/3

Note! You can use semicolon (;) in order to have multiple command at the same line

```
import numpy as np
import matplotlib.pyplot as plt

a = 0; b = 1
N = 10

x = np.linspace(a,b,N+1)

y = x**2;

y_right = y[1:]
y_left = y[:-1]

# Trapezoid Rule
dx = (b - a)/N
A = (dx/2) * np.sum(y_right + y_left)

print("A =", A)

plt.plot(x,y)
plt.xlim([0,1]); plt.ylim([0,1]);
```

SciPy

- In the previous example we implemented our own integration from scratch using the **Trapezoid** rule. That's always a good approach because then we get to know the mathematics behind.
- But typically you want to use a predefined function that do the job for you.
- These functions typically also use more advanced numerical integration methods than the simple and basic Trapezoid rule.
- SciPy has many functions for Numerical Integration
- We will use functions in the SciPy Module **scipy.integrate**
- https://docs.scipy.org/doc/scipy/reference/integrate.html

`trapz()` Function

We start with using the built-in Trapezoid rule function `trapz()`:

- `A = trapz(y,x,`Δx`)`

  x – Array of x values

  y – Array of y values

  Δx – The spacing between sample point

- https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.trapz.html#scipy.integrate.trapz

Python Code - `trapz()`

Given:

$$f(x) = x^2$$

We will find the Integral using Python:

$$\int_0^1 f(x)dx = ?$$

```
A = 0.33499999999999996
```

This is a good approximation when we now the exact answer is $A = 1/3$

```
import numpy as np

a = 0
b = 1
N = 10
dx = (b - a)/N

x = np.linspace(a,b,N+1)

y = x**2;

A = np.trapz(y,x,dx)

print(A)
```

`quad()` Function

- `A,err = quad(f, a, b)`

  f – Function that shall be integrated

  a – Lower Limit of Integration

  b – Upper Limit of Integration

- https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.quad.html#scipy.integrate.quad

Python Code – quad()

Given:

$$f(x) = x^2$$

We will find the Integral using Python:

$$\int_0^1 f(x)dx = ?$$

```
from scipy import integrate

a = 0; b = 1

def y(x):
    return x**2

A, e = integrate.quad(y, a, b)

print("A =", A)
```

This is a very good approximation when we now the exact answer is A = 1/3

```
A = 0.3333333333333337
```

SciPy – Functions Overview

There are lots of other functions you can use. I leave to you to try some of these:

Integration and ODEs (scipy.integrate)

Integrating functions, given function object

quad(func, a, b[, args, full_output, ...])	Compute a definite integral.
quad_vec(f, a, b[, epsabs, epsrel, norm, ...])	Adaptive integration of a vector-valued function.
dblquad(func, a, b, gfun, hfun[, args, ...])	Compute a double integral.
tplquad(func, a, b, gfun, hfun, qfun, rfun)	Compute a triple (definite) integral.
nquad(func, ranges[, args, opts, full_output])	Integration over multiple variables.
fixed_quad(func, a, b[, args, n])	Compute a definite integral using fixed-order Gaussian quadrature.
quadrature(func, a, b[, args, tol, rtol, ...])	Compute a definite integral using fixed-tolerance Gaussian quadrature.
romberg(function, a, b[, args, tol, rtol, ...])	Romberg integration of a callable function or method.
quad_explain([output])	Print extra information about integrate.quad() parameters and returns.
newton_cotes(rn[, equal])	Return weights and error coefficient for Newton-Cotes integration.
IntegrationWarning	Warning on issues during integration.
AccuracyWarning	

Integrating functions, given fixed samples

trapz(y[, x, dx, axis])	Integrate along the given axis using the composite trapezoidal rule.
cumtrapz(y[, x, dx, axis, initial])	Cumulatively integrate y(x) using the composite trapezoidal rule.
simps(y[, x, dx, axis, even])	Integrate y(x) using samples along the given axis and the composite Simpson's rule.
romb(y[, dx, axis, show])	Romberg integration using samples of a function.

https://docs.scipy.org/doc/scipy/reference/integrate.html

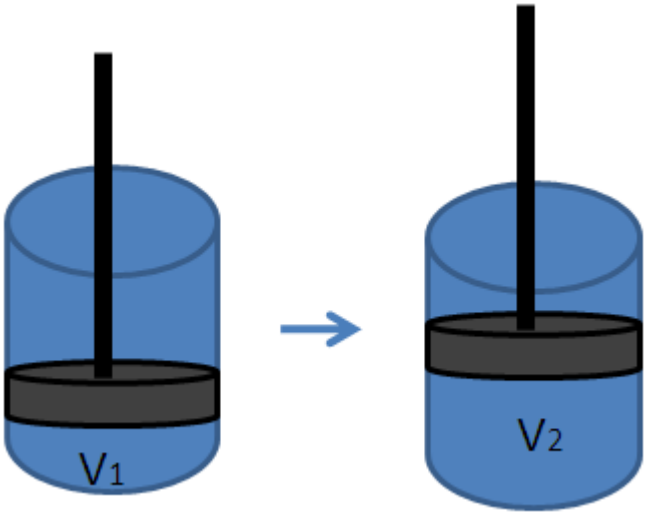
Practical Example

Hans-Petter Halvorsen

Engine Example

Given a piston cylinder device (e.g., it could be a car or a motorcycle):

We want to find the **work produced (W)** in this piston cylinder device



The amount of work (W) done is equal to the product of the force (F) exerted on the piston times the distance (d) the piston is moved: $W = F \times d$

The pressure (P) the gas exerts on the piston is equal to the force (F) with which it pushes up on the piston divided by the surface area (A) of the piston: $P = \frac{F}{A}$

This gives: $W = F \times d = (P \times A) \times d$

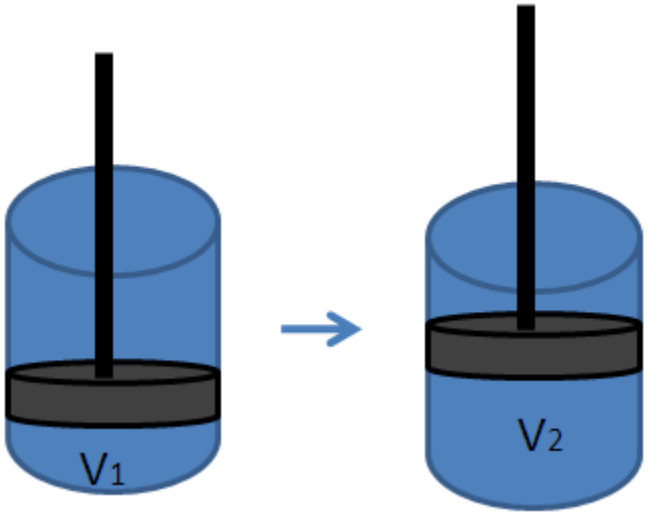
The work produced will then be: $W = P\Delta V$

Engine Example

Given a piston cylinder device (e.g., it could be a car or a motorcycle):

We want to find the **work produced (W)** in this piston cylinder device

The work produced will then be: $W = P\Delta V$



The work produced based on the change in volume from V_1 to V_2 is then:

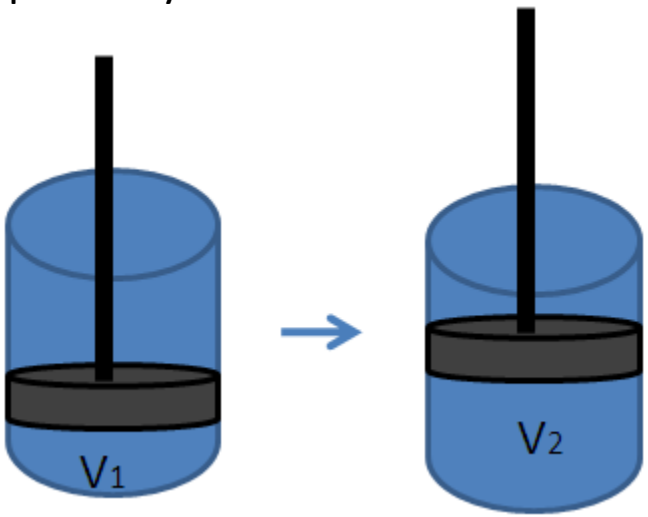
$$W = \int_{V_1}^{V_2} P dV$$

Engine Example

Given a piston cylinder device (e.g., it could be a car or a motorcycle):

We want to find the **work produced (W)** in this piston cylinder device

$$W = \int_{V_1}^{V_2} P dV$$



We can use the ideal gas law: $PV = nRT$,i.e., $P = \frac{nRT}{V}$

This gives:

$$W = \int_{V_1}^{V_2} \frac{nRT}{V} dV$$

We also assume that the piston contains 1 kmol of gas at 300K and that the temperature is constant during the process. $V_1 = 1m^3, V_2 = 5m^3$

P = Pressure
V = Volume, m^3
n = Number of moles, kmol
R = Universal gas constant, 8.314 kJ/kmol K
T = Temperature, K

Python Code

$$W = \int_{V_1}^{V_2} \frac{nRT}{V} dV$$

Parameters used:

$n = 1\ kmol$

$T = 300KR$

$= 8.314\ kJ/kmol\ K$

$V_1 = 1m^3$

$V_2 = 5m^3$

The resulting work becomes:

$$W = 4014.3kJ$$

```
from scipy import integrate

V1 = 1
V2 = 5

n = 1
R = 8.314
T = 300

def work_eq(V, n, R, T):
    W = (n*R*T)/V
    return W

W, e = integrate.quad(work_eq, V1, V2, args=(n, R, T,))

print("W =", W)
```

```
W = 4014.2600411931353
```

Integration on Polynomials

Hans-Petter Halvorsen

Polynomials

A polynomial is expressed as:

$$p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}$$

where $p_1, p_2, p_3, \ldots$ are the coefficients of the polynomial.

In Python we can use the `polyint()` function to perform integration on polynomials.
This function works the same way as the `polyder()` function which performs differentiation on polynomials.

Basic Integration Rule

$$f(x) = kx^n \quad \Rightarrow \quad \int f(x) = k\frac{1}{n+1}\,x^{n+1}$$

Examples:

$$f(x) = x \Rightarrow \int f(x) = \frac{1}{2}x^2$$

$$f(x) = x^2 \Rightarrow \int f(x) = \frac{1}{3}x^3$$

$$f(x) = 3x^2 \Rightarrow \int f(x) = 3\frac{1}{3}x^3 = x^3$$

etc.

$$f(x) = 2x^3 \Rightarrow \int f(x) = \frac{2}{4}x^4$$

Basic Integration Rule

$$f(x) = kx^n \quad \Rightarrow \quad \int_a^b f(x) = \left(k\frac{1}{n+1}\,x^{n+1}\right)\Big|_a^b$$

Examples:

$$f(x) = x \Longrightarrow \int_1^2 f(x) = \left(\frac{1}{2}x^2\right)\Big|_1^2 = \frac{1}{2}2^2 - \frac{1}{2}1^2 = 2 - \frac{1}{2} = \frac{3}{2}$$

$$f(x) = x^2 \Longrightarrow \int_0^1 f(x) = \left(\frac{1}{3}x^3\right)\Big|_0^1 = \frac{1}{3}1^2 - \frac{1}{3}0^3 = \frac{1}{3}$$

etc.

Example

Given the following polynomial:

$$p(x) = x^3 + 2x^2 - x + 3$$

The exact solution is:

$$I = \int_a^b (x^3 + 2x^2 - x + 3)dx = \left(\frac{x^4}{4} + \frac{2x^3}{3} - \frac{x^2}{2} + 3x\right)\Big|_a^b$$

$$= \frac{1}{4}(b^4 - a^4) + \frac{2}{3}(b^3 - a^3) - \frac{1}{2}(b^2 - a^2) + 3(b - a)$$

$a = -1$ and $b = 1$ gives:

$$I = \frac{1}{4}(1 - 1) + \frac{2}{3}(1 + 1) - \frac{1}{2}(1 - 1) + 3(1 + 1) = \frac{22}{3} \approx 7.33$$

Example

Given the following polynomial:

$$p(x) = x^3 + 2x^2 - x + 3$$

Note!!! In order to use it in Python, we reformulate:

$$p(x) = 3 - x + 2x^2 + x^3$$

The Indefinite Integral is:

$$\int p(x) = 3x - \frac{1}{2}x^2 + \frac{2}{3}x^3 + \frac{1}{4}x^4$$

The Definite Integral is:

$$\int_{-1}^{1} p(x) = \left(3x - \frac{1}{2}x^2 + \frac{2}{3}x^3 + \frac{1}{4}x^4\right)\Big|_{-1}^{1} = \frac{22}{3} \approx 7.33$$

Python

$$p(x) = 3 - x + 2x^2 + x^3$$

```
import numpy.polynomial.polynomial as poly

p = [3, -1, 2, 1]

# Find Indefinite integral
I = poly.polyint(p)
print("I =", I)

# Find Definite integral
a = -1
b = 1

A = poly.polyval(b, I) - poly.polyval(a, I)
print("A =", A)
```

The Indefinite Integral is:

$$\int p(x) = 3x - \frac{1}{2}x^2 + \frac{2}{3}x^3 + \frac{1}{4}x^4$$

The Definite Integral is:

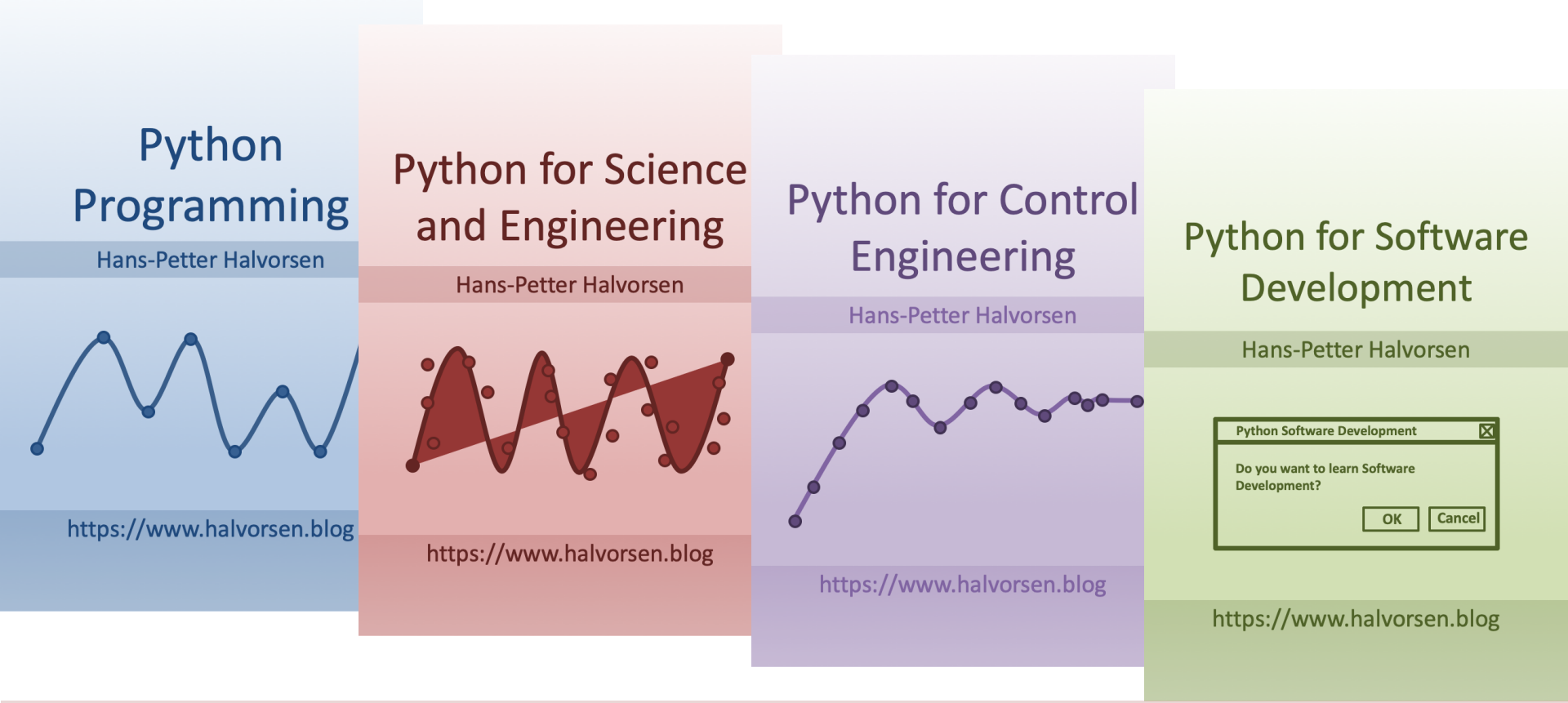
$$\int_{-1}^{1} p(x) = \left(3x - \frac{1}{2}x^2 + \frac{2}{3}x^3 + \frac{1}{4}x^4\right)\Big|_{-1}^{1}$$

$$= \frac{22}{3} \approx 7.33$$

The Python solution:

```
I = [ 0.   3.   -0.5   0.66666667   0.25 ]
A = 7.333333333333333
```

Additional Python Resources



https://www.halvorsen.blog/documents/programming/python/

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog