Scientific Computing in Python

Special Functions : scipy.special

Special functions are transcendental functions.
Using special functions in Scipy, mathematical operations can be performed on data. Various mathematical special functions are important in many computational physics problems. SciPy provides an implementation of a very rich set of special features.
SciPy provides several special functions that are used in mathematical physics , such as elliptic, convenience functions, gamma, beta, etc.
Some most frequently used special functions −

- Cubic Root Function
- Exponential Function
- Relative Error Exponential Function
- Log Sum Exponential Function
- Lambert Function
- Permutations and Combinations Function
- Gamma Function
- Bessel Function

Example 1: Program to find the cube root

```
from scipy.special import cbrt

# cube root of elements in an array

arr = [8, 512, 864, 64]

arr = list(map(cbrt,arr))

print(arr)
```

Output : [2.0, 8.0, 9.524406311809196, 4.0]

Scientific Computing in Python

Example 2 : Program to find the powers of 10 for a range

```
from scipy.special import exp10
# exponent raise to power 10 for a range
for i in range(1,5):
  print(exp10(i))
```

Output:

```
10.0
100.0
1000.0
10000.0
```

Example 3 : Program to find combination of the given numbers

```
from scipy.special import comb
print(comb(12, 3, exact = False,repetition=True))
```

Output : `364.0`

Example 4 : Program to find permutation of the given numbers

```
# import permutations module
from scipy.special import perm
# permutations of 7
print([perm(7, 1), perm(7, 2), perm(7, 3),
  perm(7, 4), perm(7, 5)])
```

Output : `[7.0, 42.0, 210.0, 840.0, 2520.0]`

Scientific Computing in Python

Example 5 : Bessel Function

Bessel functions are used in wave propagation problems in physics .

The following is an example of Bessel's function

```
from scipy.special import jn, yn, jn_zeros, yn_zeros

n = 0 # order

x = 0.0

# Bessel function of first kind

print ("J_%d(%f) = %f" % (n, x, jn(n, x)))

x = 1.0

# Bessel function of second kind

print ("Y_%d(%f) = %f" % (n, x, yn(n, x)))
```

Output:

```
J_0(0.000000) = 1.000000

Y_0(1.000000) = 0.088257
```