Basic Calculations

Algebra and Counting

In [ ]:
```
%reset -f

from sympy import *
from sympy.stats import *
from sympy.functions import *
from sympy.functions.combinatorial.numbers import *
from sympy.integrals import *
```

In [ ]:
```
# Symbols

theta = Symbol('theta', positive=True)
x = Symbol('x')
n = Symbol('n', positive=True)
t = Symbol('t', positive=True)
```

Combinatorics

In [ ]:
```
# nCr

nC(3,2)
```

In [ ]:
```
# nPr

nP(3,2)
```

In [ ]:
```
# factorial

factorial(5)
```

In [ ]:
```
# free space
```

Expected Value, Variance and Moments

In [ ]:
```
%reset -f

from sympy import *
from sympy.stats import *
from sympy.functions import *
from sympy.integrals import *
```

In [ ]:
```
# Symbols

theta = Symbol('theta', positive=True)
x = Symbol('x')
n = Symbol('n', positive=True)
t = Symbol('t', positive=True)
```

In [ ]:
```
# density

# UNCOMMENT here to use an inbuilt distribution
#X = Normal('x', 0, theta)
```

```
#lhood = simplify(density(X)(x))

# UNCOMMENT here to define manually
pdf = theta**x*(1-theta)**(1-x)

pdf
```

In [ ]:
```
# MUST RUN BEFORE VARIANCE/MOMENTS
# expected value
# also works for g(x) transformed RV

def expected_value(of, pdf, wrt, lower, upper):
    return integrate(of*pdf, (wrt, lower, upper), conds='none').doit()

# what you're finding expected value of i.e. x, 1/x, x**2, etc
of = x

# define variable and bounds
wrt = x
lower = 0
upper = 1

exp_val = expected_value(of, pdf, wrt, lower, upper)
exp_val = simplify(exp_val)
exp_val
```

In [ ]:
```
# variance

variance = expected_value(wrt**2, pdf, wrt, lower, upper) - exp_val**2
variance = simplify(variance)
variance
```

In [ ]:
```
# rth raw moment

r = 3

raw_moment = expected_value(wrt**r, pdf, wrt, lower, upper)
raw_moment = simplify(raw_moment)
raw_moment
```

In [ ]:
```
# rth central moment
# uses the risch algorithm to allow integration by parts

r = 1

integrand = ((wrt - exp_val)**r)*pdf
cent_moment = integrate(integrand, wrt, risch=True, conds='none').doit()
cent_moment = simplify(cent_moment)
cent_moment
```

In [ ]:
```

```

Calculus

In [ ]:
```
%reset -f

from sympy import *
from sympy.stats import *
from sympy.functions import *
from sympy.integrals import *
```

In [ ]:
```
# Symbols

theta = Symbol('theta', positive=True)
x = Symbol('x', positive=True)
n = Symbol('n', positive=True)
t = Symbol('t', positive=True)
```

In [ ]:
```
# free space
```

Derivative

In [ ]:
```
# first derivative of f

f = x**2 - 2*x + 1
wrt = x

diff1 = diff(f, wrt)
diff1
```

In [ ]:
```
# second derivative of f

wrt = x

diff2 = diff(diff1, wrt)
diff2
```

In [ ]:
```
# free space
```

Integral

In [ ]:
```
# indefinite integral of f
# automatically has the risch algorithm available in case of integration by parts

f = x**2 - 2*x + 1
wrt = x

integrate(f, wrt, conds='none', risch=True).doit()
```

In [ ]:
```
# definite integral of f

f = x**2 - 2*x + 1

wrt = x
lower = 0
upper = 10

integrate(f, (wrt, lower, upper), conds='none').doit()
```

In [ ]:
```
# indefinite integral of the product of two expressions

f = theta**2
g = exp(-theta/(x))
wrt = theta

integrate(f*g, wrt, conds='none', risch=True).doit()
```

In [ ]:
```
# definite integral of the product of two expressions

f = theta**2
g = exp(-theta/(x+5))
```

```
wrt = theta
lower = 0
upper = oo

integrate(f*g, (wrt, lower, upper), conds='none').doit()
```

In [ ]:
```
# plot if needed

plot(f, xlim=(0, 10), ylim=(0, 20), adaptive=False, nb_of_points=500)
```

Free Calculation Space

In [ ]:
```

```

In [ ]:
```

```