数学:解析学

関数

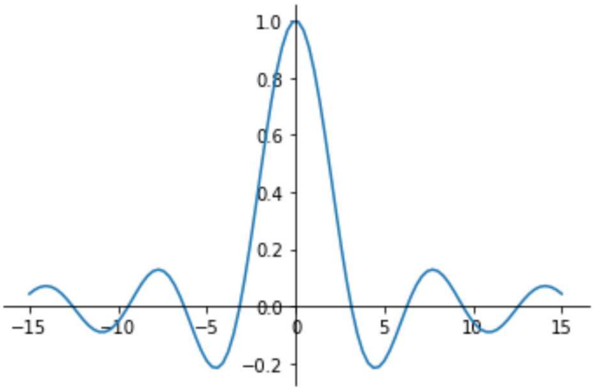
In [ ]:
```
${y=f(x)}={\frac{sin(x)}{x}}$

Domain : $(-\infty,+\infty)-\{0\}$ or $\forall x \in R: x\ne 0$

Range: $[-0.21,1]$

$\lim_{x \to 0} f(x)=1$

Similar to other functions:

${y=f(x)}={2\frac{sin(x)}{x}}$

${y=f(x)}={3\frac{sin(x)}{x}}$
```

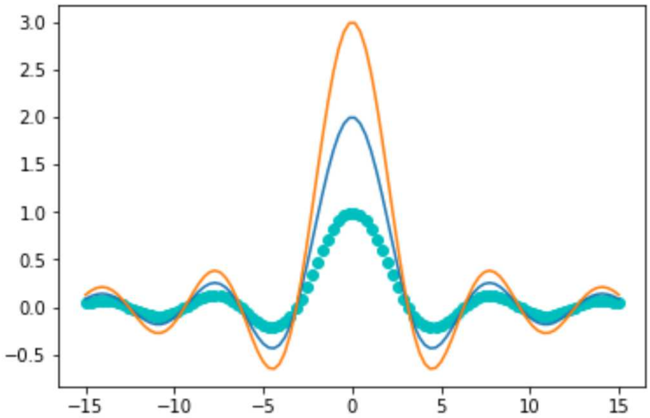
In [0]:
```python
import pylab
import numpy
```

In [0]:
```python
def my_plot(x,y):
  ax = pylab.gca()   # gca stands for 'get current axis'
  ax.spines['right'].set_color('none')
  ax.spines['top'].set_color('none')
  ax.xaxis.set_ticks_position('bottom')
  ax.spines['bottom'].set_position(('data',0))
  ax.yaxis.set_ticks_position('left')
  ax.spines['left'].set_position(('data',0))

  pylab.plot(x,y)
  pylab.show()
```

In [0]:

```
x = numpy.linspace(-15,15,100) # 100 linearly spaced numbers
y = numpy.sin(x)/x # computing the values of sin(x)/x

# compose plot
#pylab.axis('normal')
my_plot(x,y) # sin(x)/x
pylab.plot(x,y,'co') # same function with cyan dots
pylab.plot(x,2*y,x,3*y) # 2*sin(x)/x and 3*sin(x)/x
pylab.show() # show the plot
```



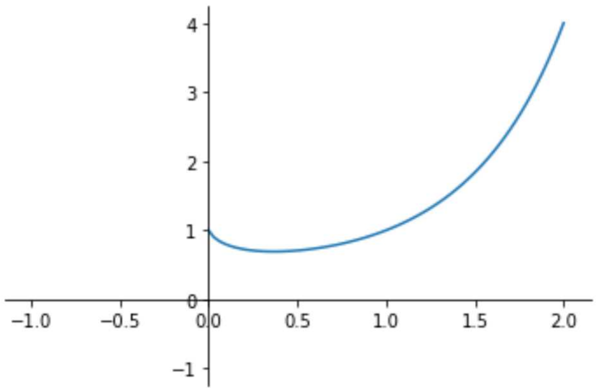


In [ ]:

```
${y=f(x)}={{x}^{x}}$

Domain : $(0,+\infty)$

Range: $(1,+\infty)$

$\lim_{x \to 0} f(x)$
```

In [0]:
```
x = numpy.linspace(-1,2,100) # 100 linearly spaced numbers
y = x**x

my_plot(x,y) # x^x
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2: RuntimeWarning
: invalid value encountered in power
```



極限値計算のプログラム

In [0]:
```
def f(x):
  return x**x

def limit(f,val):
  return int(f(val))

print("limit f(x), x->0 is ",limit(f,0))
```

```
limit f(x), x->0 is  1
```

In [0]:
```
import numpy

def f(x):
  return numpy.sin(x)/x

print("limit f(x), x->0 is ",limit(f,10**-20))
```

```
limit f(x), x->0 is  1
```

Pythonライブラリ(sympy)でシンボル的な数学(Symbolic Math with Python sympy library)

In [27]:
```
# try it out:
import sympy
sympy.init_printing()
x = sympy.symbols('x')
```

In [0]:
```
from IPython.display import Math, HTML

def load_mathjax_in_cell_output():
  display(HTML("<script src='https://www.gstatic.com/external_hosted/"
               "mathjax/latest/MathJax.js?config=default'></script>"))
get_ipython().events.register('pre_run_cell', load_mathjax_in_cell_output)
```

In [0]:
```
print("limit of f(x)=x^x x->0 is:")
sympy.limit(x**x,x,0)
```

```
limit of f(x)=x^x x->0 is:
```

Out[0]: 1

In [0]:
```
sympy.limit(sympy.sin(x)/x,x,0)
```

Out[0]: 1

In [0]:
```
#sympy.latex(sympy.limit(sympy.sin(x)/x,x,0))
```

Out[0]: '1'

In [0]:
```
f=sympy.Function('f')
f=(sympy.sqrt(x+1)-sympy.sqrt(x))*sympy.sqrt(x)
f
```

Out[0]: $\sqrt{x}\left(-\sqrt{x}+\sqrt{x+1}\right)$

In [0]:
```
sympy.limit(f,x,sympy.oo)
```

Out[0]: $\frac{1}{2}$

In [0]:
```
f=sympy.Function('f')
f=(sympy.sqrt(x+1)-sympy.sqrt(x))
f
```

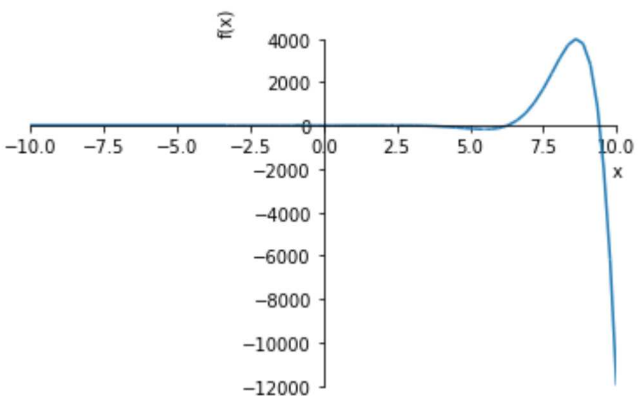
Out[0]: $-\sqrt{x}+\sqrt{x+1}$

In [0]:
```
print("limit f(x), x->Infinity:")
sympy.limit(f,x,sympy.oo)
```

```
limit f(x), x->Infinity:
```

Out[0]: 0

In [0]:
```
import sympy.plotting.plot as splt
#x1 = numpy.linspace(-1,2,100) # 100 linearly spaced numbers
p1=splt(f,show=False)
p1.show()
```



In [0]:
```
# find the minimum or maximum of a function by solving f'(x)=0
df = f.diff(x)
sln=sympy.solve(sympy.simplify(df),x)
print(sln)
```

```
[zoo, -pi/4, 3*pi/4]
```

In [0]:
```
min_max=[]
for i in range(0,4):
  min_max.append(((-1/4+i)*numpy.pi,f.subs(x,(-1/4+i)*numpy.pi))) #.evalf()
min_max
```

Out[0]: $[(-0.7853981633974483, \quad -0.322396941944834), \quad (2.356194490192345, \quad 7.46048853929$
$(5.497787143782138, \quad -172.640872178161), \quad (8.63937979737193, \quad 3995.0293589297$

$f(3\frac{\pi}{4})$, $f'(3\frac{\pi}{4}+2\pi)$,$f(3\frac{\pi}{4}+2\pi)$

In [0]:
```
f.subs(x,3*numpy.pi/4)
```

Out[0]: 7.4604885392934

In [0]:
```
3*numpy.pi/4+2*numpy.pi
```

Out[0]: 8.63937979737193

In [0]:
```
df.subs(x,3*numpy.pi/4+2*numpy.pi)
```

Out[0]: $9.54969436861575 \cdot 10^{-12}$

In [0]:
```
f.subs(x,3*numpy.pi/4+2*numpy.pi)
```

Out[0]: 3995.02935892975

In [0]:

In [ ]:
```
##関数の連続性(Continuity of a function)

* 定義１：

関数$f(x)$が$x=a$において連続であるとは$lim_{x->a} f(x) = f(a)$が成り立つことである

* 定義2：

関数$f(x)$が区間$I$で連続とは、区間$I$の各点で連続になることである

* 定理１：

関数$f(x)$が$x=a$で連続、$z=g(y)$が$y=f(a)$で連続のとき、合成関数$z=g(f(x)=g\circ f(x
)$は$x=a$で連続である

* 定理１：

関数$f(x)$が$x=a$で連続、$z=g(y)$が$y=f(a)$で連続のとき、合成関数$z=g(f(x)=g\circ f(x
)$は$x=a$で連続である

* 定理２：

関数$y=f(x)$が$閉区間[a,b]$において連続で、$f(a)\ne f(b)$ならば、$f(x)$は$f(a)$と$f(b
)$の中間の任意の値をとる。

* 定理３：

関数$y=f(x)$が$閉区間[a,b]$において連続ならば、$f(x)$有界であり、$[a,b]$において最大値、
最小値をとる。
```

In [ ]:
```
指数関数$y=e^x$は連続かつ狭義の単調増加である。逆関数が自然対数$log(x)$で、定理より
連続かつ狭義単調増加関数になる。

（1）$lim_{x->0} \frac{log(1+x)}{x}$

（2）$lim_{x->0} \frac{e^x-1}{x}$
```

関数の微分法

In [ ]:
```
##導関数

変化を科学的に扱うには何らかの量を関数として表す必要がある。その関数の変化の様子を詳しく調べるの
が「微分」である。変化は差をとるということである。
ＧＰＳによりリアルタイムで自分の位置を正解に知ることができる。

*　定義 1:

区間$I$の点$x=a$で、極限値 $lim_{x->a} \frac{f(x)-f(a)}{x-a}$ が存在する時、この極限値
を$f'(a)$　と書き、$a$における$f(x)$の微分係数(differential coefficient)または微分商(di
fferential quotient)という。またこのとき、$f(x)$は$x=a$で微分可能(differentiable)である
という。

*　定義　2：

関数$f(x)$が区間$I$の各点で微分可能なとき、極限値 $lim_{h->0} \frac{f(x＋ｈ)-f(ｘ)}{ｈ}
$ が$f'(x)$　と書き、$f(x)$の導関数(derived function)という。
```

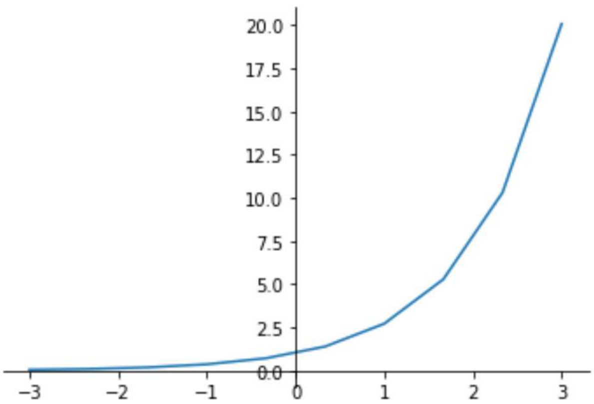
In [28]:
```
import sympy
f=sympy.Function('f')
h = sympy.symbols('h')
x = sympy.symbols('x')
f=sympy.exp(x)
d =(sympy.exp(x+h)-sympy.exp(x))/h
d
```

Out[28]: $\frac{1}{h}\left(-e^{x}+e^{h+x}\right)$

In [29]:
```
# compute the limit when h->0
sympy.limit(d,h,0)
```

Out[29]: e^{x}

In [21]:
```
x = numpy.linspace(-3,3,10) # 100 linearly spaced numbers
y = numpy.exp(x)
# compose plot
my_plot(x,y) #
```



In [30]:
```
f.diff()
```

Out[30]: e^{x}

In [0]:
```
f=sympy.Function('f')
x = sympy.symbols('x')
f=sympy.sin(x)*sympy.exp(x)
f
```

Out[0]: $e^{x}\sin\left(x\right)$

In [0]:
```
#f'(x)    # sympy.init_printing(use_unicode=True)
f.diff(x)
```

Out[0]: $e^{x}\sin\left(x\right)+e^{x}\cos\left(x\right)$

In [0]:
```
# try it out:
#import sympy
#sympy.init_printing()
#x = sympy.symbols('x')
#sympy.Integral(sympy.sqrt(1 / x), x)
```