Table of Contents

線形代数

グラムシュミットの直交化

In [19]:
```
from sympy import *

init_printing()
x1 = Matrix([1,1,1])
x2 = Matrix([0,1,0])
x3 = Matrix([-1,1,0])
```

In [135]:
```
GramSchmidt([x1,x2,x3])
```

Out[135]:
$$\left[\begin{bmatrix}1\\1\\1\end{bmatrix}, \begin{bmatrix}-\frac{1}{3}\\\frac{2}{3}\\-\frac{1}{3}\end{bmatrix}, \begin{bmatrix}-\frac{1}{2}\\0\\\frac{1}{2}\end{bmatrix}\right]$$

In [141]:
```
a1 = x1/x1.norm()
a1
```

Out[141]:
$$\begin{bmatrix}\frac{\sqrt{3}}{3}\\\frac{\sqrt{3}}{3}\\\frac{\sqrt{3}}{3}\end{bmatrix}$$

In [149]:
```
y2 = x2-x2.dot(a1)*a1
```

In [150]:
```
a2 = y2/y2.norm()
a2
```

Out[150]:
$$\begin{bmatrix}-\frac{\sqrt{6}}{6}\\\frac{\sqrt{6}}{3}\\-\frac{\sqrt{6}}{6}\end{bmatrix}$$

In [154]:
```
y3 = x3 - x3.dot(a1)*a1-x3.dot(a2)*a2
y3
```

Out[154]:
$$\begin{bmatrix}-\frac{1}{2}\\0\\\frac{1}{2}\end{bmatrix}$$

In [155]:
```
a3 = y3/y3.norm()
a3
```

Out[155]:
$$\begin{bmatrix}-\frac{\sqrt{2}}{2}\\0\\\frac{\sqrt{2}}{2}\end{bmatrix}$$

直交補空間

In [32]:
```
x1,x2,x3,a,b=symbols('x1,x2,x3,a,b')
eq1=2*x1-x2+x3
eq2=x1-3*x2+x3

solve({eq1,eq2},{x1,x2,x3})
```

Out[32]:
$$\left\{x_1 : -\frac{2x_3}{5}, \quad x_2 : \frac{x_3}{5}\right\}$$

In [33]:
```
eq3 = expand(5/a*(-Rational(2,5)*a*x1+Rational(1,5)*a*x2+a*x3))
```

In [34]:
```
solve(eq3.subs({x2:a,x3:b}),x1)
```

Out[34]:
$$\left[\frac{a}{2} + \frac{5b}{2}\right]$$

In [35]:
```
a*Matrix([Rational(1,2),1,0]) + b*Matrix([Rational(5,2),0,1])
```

Out[35]:

$$\left[\begin{matrix}\frac{a}{2} + \frac{5b}{2}\\a\\b\end{matrix}\right]$$

微積分

Taylor展開

In [10]:
```
from sympy import *
init_session()

t = symbols('t')
v = exp(-t)+1.0
```

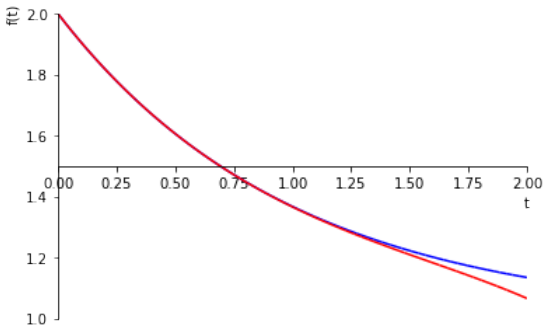
```
IPython console for SymPy 1.0 (Python 3.6.1-64-bit) (ground types:
python)

These commands were executed:
>>> from __future__ import division
>>> from sympy import *
>>> x, y, z, t = symbols('x y z t')
>>> k, m, n = symbols('k m n', integer=True)
>>> f, g, h = symbols('f g h', cls=Function)
>>> init_printing()

Documentation can be found at http://docs.sympy.org/1.0/
```

In [11]:
```
vs = v.series(t,0,6)
vs
```

Out[11]:

$$2.0 - t + \frac{t^2}{2} - \frac{t^3}{6} + \frac{t^4}{24} - \frac{t^5}{120} + \mathcal{O}\left(t^6\right)$$

In [14]:
```
%matplotlib inline
vsO = vs.removeO()
p = plot(v, vsO, (t,0,2), ylim=[1,2], show=False)
p[0].line_color = 'b'
p[1].line_color = 'r'
p.show()
```



積分の比較

In [15]:
```
i_v = integrate(v,(t,0,2))
pprint(i_v)
i_v.evalf()
```

```
        -2
- 1.0⋅e    + 3.0
```

Out[15]: 2.86466471676339

In [16]:
```
i_vs = integrate(vsO,(t,0,2))
pprint(i_vs)
i_vs.evalf()
```

```
2.84444444444444
```

Out[16]: 2.84444444444444

積分の誤差

誤差をわかりやすくするには，下のようにまとめれば良い．そうすると必要な次数は7(8)次であることがわかる．

In [17]:
```
vs = v.series(t,0,10)
pprint(vs)
vs0 = vs.removeO()
i_vs = integrate(vs0,(t,0,2))
pprint(i_vs)
i_vs.evalf()-i_v.evalf()
```

```
              2     3     4      5      6      7        8         9          ⎛ 1
             t     t     t      t      t      t        t         t           ⎝
0⎞
2.0 - t + ── - ── + ── - ─── + ─── - ──── + ───── - ────── + O⎝t
⎠
             2     6     24    120    720    5040    40320    362880
2.86462081128748
```

Out[17]: $-4.39054759100443 \cdot 10^{-5}$

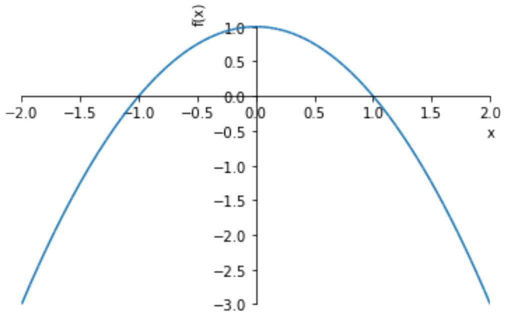
1-(2) 図形の面積

2点Q(x_1, y_1)とR(x_2, y_2)を通る直線の方程式は

$$y - y_1 = \frac{y_1 - y_2}{x_1 - x_2}(x - x_1)$$

で求められる.

In [72]:
```
from sympy import *
init_printing()
b, x, t = symbols('b,x,t')
y_c = 1-x**2
plot(y_c,(x,-2,2))
```



Out[72]: <sympy.plotting.plot.Plot at 0x112dec438>

In [73]:
```
s_c = solve(y_c,x)
s_c
```

Out[73]: $[-1, \quad 1]$

In [74]:
```
x1=s_c[0]
y1=0
x2=s_c[1]-b
y2=y_c.subs({x:x2})
```

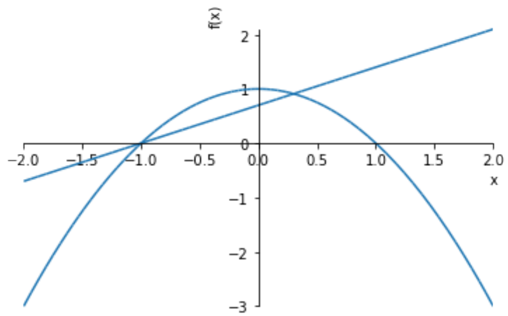
In [75]:
```
m = (y1-y2)/(x1-x2)
simplify(m)
```

Out[75]: b

In [76]:
```
y_m = expand(simplify(m)*(x+1))
y_m
```

Out[76]: $bx + b$

In [77]:
```
plot(y_c,y_m.subs({b:0.7}),(x,-2,2))
```



Out[77]: <sympy.plotting.plot.Plot at 0x11cee22e8>

In [78]:
```
s1=expand(integrate(y_c-y_m,(x,-1,1-b)))
s1
```

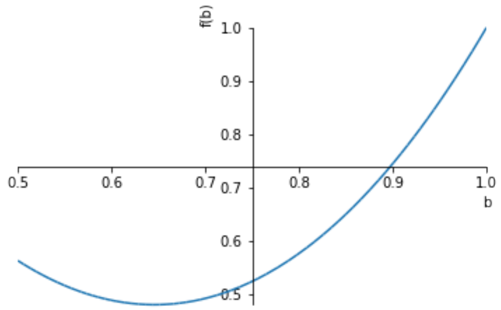
Out[78]: $-\frac{b^3}{6} + b^2 - 2b + \frac{4}{3}$

In [79]:
```
s2=expand(integrate(y_m-y_c,(x,1-b,b)))
s2
```

Out[79]: $\frac{2b^3}{3} + 2b^2 - \frac{5b}{2} + \frac{2}{3}$

In [80]:
```
expand(s1+s2)
```
Out[80]:
$$\frac{b^3}{2} + 3b^2 - \frac{9b}{2} + 2$$

In [81]:
```
plot(s1+s2,(b,0.5,1))
```



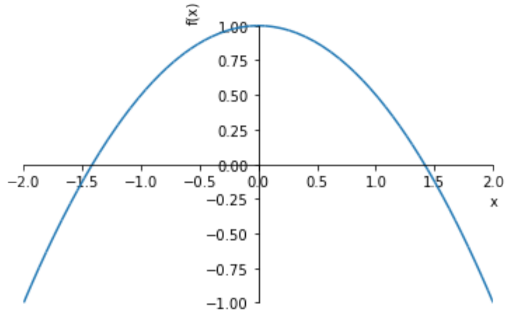
Out[81]: <sympy.plotting.plot.Plot at 0x119885438>

In [82]:
```
solve(diff(s1+s2,b),b)
```
Out[82]:
$$\left[-2+\sqrt{7}, \quad -\sqrt{7}-2\right]$$

1-(2)改

放物線Cの方程式を$y = 1 - 0.5x^2$として問題を解く． 放物線C上の2点Q($-\sqrt{2}, 0$)とR($\sqrt{2} - b, 1 - (\sqrt{2} - b)^2$)と読み替える．また，$S_2$を求めるときの範囲は$\sqrt{2} - b \leq x \leq b$と読み替える．

In [2]:
```
from sympy import *
init_printing()
b, x, t = symbols('b,x,t')
# y_c = 1-0.5*x**2
y_c = 1-Rational(1,2)*x**2
plot(y_c,(x,-2,2))
```



Out[2]: <sympy.plotting.plot.Plot at 0x115d64c88>

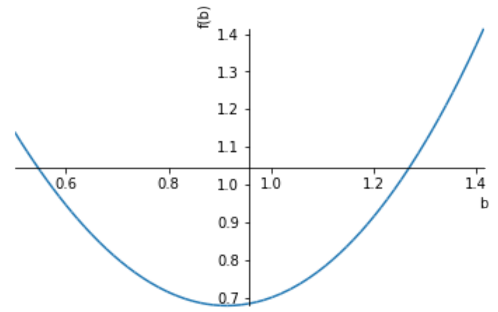
In [3]:
```
s_c = solve(y_c,x)
s_c
```
Out[3]:
$$\left[-\sqrt{2}, \quad \sqrt{2}\right]$$

In [4]:
```
x1=s_c[0]
y1=0
x2=s_c[1]-b
y2=y_c.subs({x:x2})
```

In [10]:
```
y_m = simplify((y1-y2)/(x1-x2)*(x-x1)+y1)
y_m
```
Out[10]:
$$\frac{\left(x + \sqrt{2}\right)\left(\left(b - \sqrt{2}\right)^2 - 2\right)}{2b - 4\sqrt{2}}$$

In [11]:
```
s1= expand(integrate(y_c-y_m,(x,x1,x2)))
simplify(s1)
```
Out[11]:
$$-\frac{1}{12b - 24\sqrt{2}}\left(b^4 - 8\sqrt{2}b^3 + 48b^2 - 64\sqrt{2}b + 64\right)$$

In [12]:
```
s2=expand(integrate(y_m-y_c,(x,x2,b)))
s2
```
Out[12]:
$$\frac{b^3}{3} + \frac{2\sqrt{2}b^3}{2b-4\sqrt{2}} - \frac{14b^2}{2b-4\sqrt{2}} + \frac{b}{2} + \frac{14\sqrt{2}b}{2b-4\sqrt{2}} - \frac{\sqrt{2}}{3} - \frac{8}{2b-4\sqrt{2}}$$

In [13]:
```
plot(s1+s2,(b,1/2,s_c[1]))
```


Out[13]: <sympy.plotting.plot.Plot at 0x10da6cef0>

In [14]:
```
solve(diff(s1+s2,b),b)
```
Out[14]: []

In [15]:
```
solve(diff(s1+s2,b),b)[1].evalf()
```
```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent c
all last)
<ipython-input-15-3fb892d5818b> in <module>()
----> 1 solve(diff(s1+s2,b),b)[1].evalf()

IndexError: list index out of range
```

In [16]:
```
solve(simplify(diff(s1+s2,b)),b)
```
Out[16]: $\left[-\sqrt{14}-2\sqrt{2},\quad -2\sqrt{2}+\sqrt{14}\right]$

In [17]:
```
solve(simplify(diff(s1+s2,b)),b)[1].evalf()
```
Out[17]: 0.913230262027751

与関数を

$$y = 1 - \frac{1}{2}x^2$$

つまり

```
y=1-Rational(1,2)*x**2
```

とRationalを明示的に使えば，答えは，

$$\left[-\sqrt{14}-2\sqrt{2},\quad -2\sqrt{2}+\sqrt{14}\right]$$

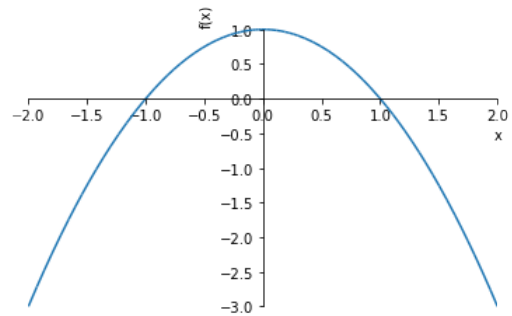
と解析的に求められる．後ろ側が求めた数値解と一致する． その場合，

```
solve(simplify(diff(s1+s2,b)),b)
```

としないと求められない．

1-(1) 放物線の接線の距離

2015 年度大学入試センター試験 追試 数学 II・B 第 2 問 (1)の解答例を参考に示しておく(苦労して解いたんで).

In [47]:
```
from sympy import *
init_printing()
a, x, t = symbols('a,x,t')
y_c = 1-x**2
plot(y_c,(x,-2,2))
```


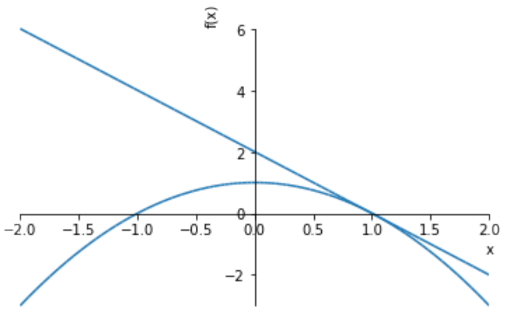
Out[47]: <sympy.plotting.plot.Plot at 0x10a3d8748>

In [48]:
```
m = diff(y_c,x)
m
```
Out[48]: $-2x$

In [49]:
```
y_l=collect(expand(m.subs({x: a})*(x-a)+y_c.subs({x: a})),x)
y_l
```

Out[49]: $a^2 - 2ax + 1$

In [50]:
```
plot(y_l.subs({a:1}),y_c,(x,-2,2))
```



Out[50]: <sympy.plotting.plot.Plot at 0x114b0f0b8>

点(x_0, y_0)と直線$(c_a x + c_b y + c_c = 0)$の距離$(h)$の公式

$$h = \frac{|c_a x_0 + c_b y_0 + c_c|}{\sqrt{c_a^2 + c_b^2}}$$

In [51]:
```
c_a=y_l.coeff(x)
c_c=y_l.coeff(x,0)
c_b=-1
```

In [52]:
```
h = (c_a*0+c_b*0+c_c)/sqrt(c_a**2+c_b**2)
h
```

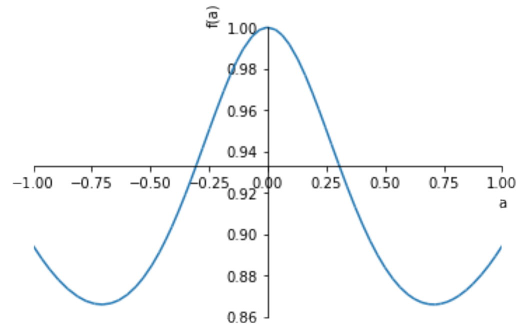
Out[52]: $\frac{a^2 + 1}{\sqrt{4a^2 + 1}}$

In [53]:
```
a_2 = solve(sqrt(4*a**2+1)**2-t**2,a**2)[0]
```

In [54]:
```
simplify((a_2+1)/t)
```

Out[54]: $\frac{t^2 + 3}{4t}$

In [55]:
```
plot(h,(a,-1,1))
```



Out[55]: <sympy.plotting.plot.Plot at 0x114b64160>

In [56]:
```
s1 = solve(diff(h,a),a)
s1
```

Out[56]: $\left[0, \quad -\frac{\sqrt{2}}{2}, \quad \frac{\sqrt{2}}{2}\right]$

In [57]:
```
h.subs({a:s1[1]})
```

Out[57]: $\frac{\sqrt{3}}{2}$