66310837

นายจิรัฐ ฟองดา

```
# conda activate nm เปิด env
import numpy as np
import matplotlib.pyplot as plt
from math import factorial
import sympy as sp
```

```
print(100*(0.1 + 0.2))
print(100*0.1 + 100*0.2)
```

```
30.000000000000004
30.0
```

```
100*(0.1 + 0.2) == 100*0.1 + 100*0.2
```

```
False
```

```
(np.pi+1e100)-1e100
```

```
0.0
```

```
(np.pi)+(1e100-1e100)
```

```
3.141592653589793
```

```
x = 1.48234
y = 1.48235
```

```
x_db1 = np.float64(x)
y_db1 = np.float64(y)
diff_db1 = x_db1-y_db1

diff_db1
```

```
np.float64(-1.0000000000065512e-05)
```

```
x_sng = np.float32(x)
y_sng = np.float32(y)
diff_sng = x_sng-y_sng
```

```
print(diff_sng)
```

```
-1.001358e-05
```

```
sp.init_printing()
sp.var("x")
x
```

x

```
g = sp.sin(sp.sqrt(x)+2)**2
g
```

$\sin^{2}\left(\sqrt{x}+2\right)$

```
g.diff(x)
```

$\sin\left(\sqrt{x}+2\right)\cos\left(\sqrt{x}+2\right)/\sqrt{x}$

```
g.diff(x, 4)
```

$$\frac{\sin^{2}\left(\sqrt{x}+2\right)}{2x^{2}}-\frac{\cos^{2}\left(\sqrt{x}+2\right)}{2x^{2}}-\frac{15\sin^{2}\left(\sqrt{x}+2\right)}{8x^{3}}+\frac{15\cos^{2}\left(\sqrt{x}+2\right)}{8x^{3}}+\frac{3\sin\left(\sqrt{x}+2\right)\cos\left(\sqrt{x}+2\right)}{x^{5/2}}-\frac{15\sin\left(\sqrt{x}+2\right)\cos\left(\sqrt{x}+2\right)}{8x^{7/2}}$$

```
g.subs(x, 1)
```

$\sin^{2}\left(3\right)$

```
g.subs(x, 1).evalf()
```

0.019914856674817

```
def plot_sympy(my_f, my_pts, **kwargs):
    f_values = np.array([my_f.subs(x, pt) for pt in my_pts])
    plt.plot(pts, f_values, **kwargs)

f = 1/(20*x-10)

f
```

$\frac{1}{20x-10}$

```
f.diff(x)
```

$-\frac{20}{\left(20x-10\right)^{2}}$

```
f.diff(x, 2)
```

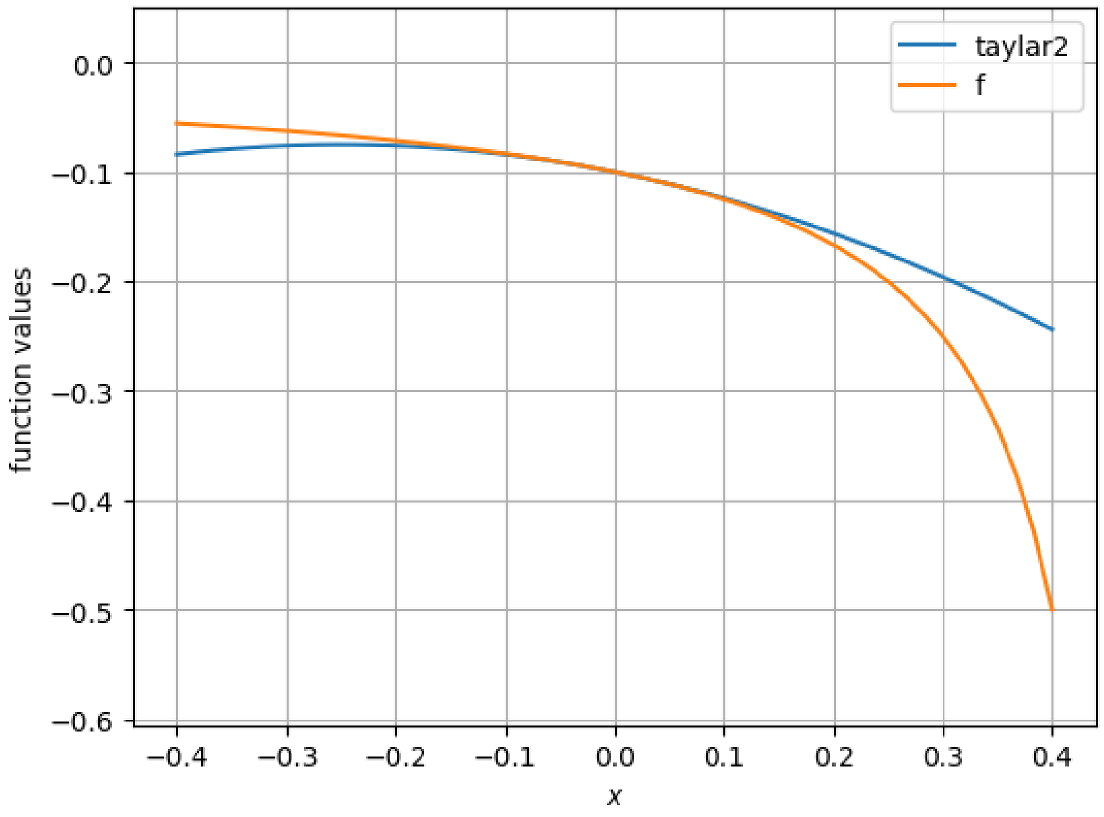
$\frac{4}{5\left(2x-1\right)^{3}}$

```python
taylor2 = (
    f.subs(x, 0)
    + f.diff(x).subs(x, 0) * x
    + f.diff(x, 2).subs(x, 0)/2 * x**2
)

taylor2
```

$-2x^2/5 - x/5 - 1/10$

```python
pts = np.linspace(-0.4, 0.4)

plot_sympy(taylor2, pts, label="taylar2")
plot_sympy(f, pts, label="f")
plt.legend(loc="best")
plt.axis("equal")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("function values")
```

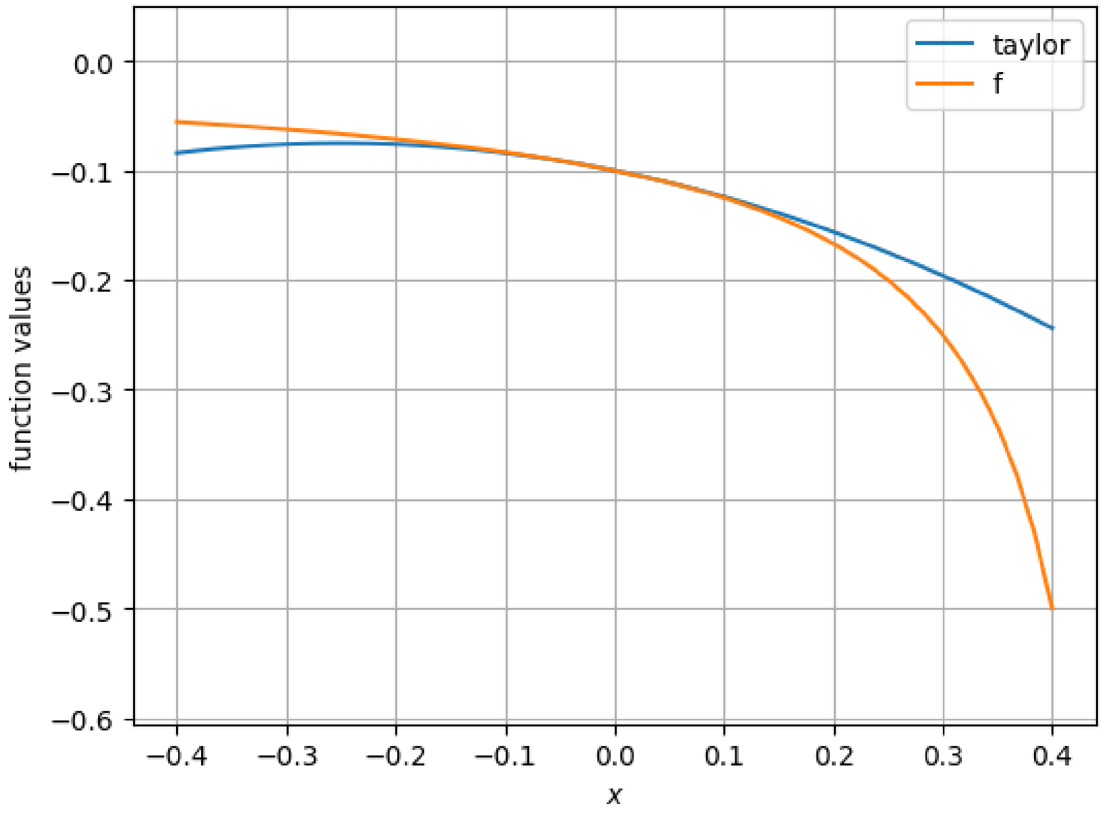
```
Text(0, 0.5, 'function values')
```



```python
n = 2
```

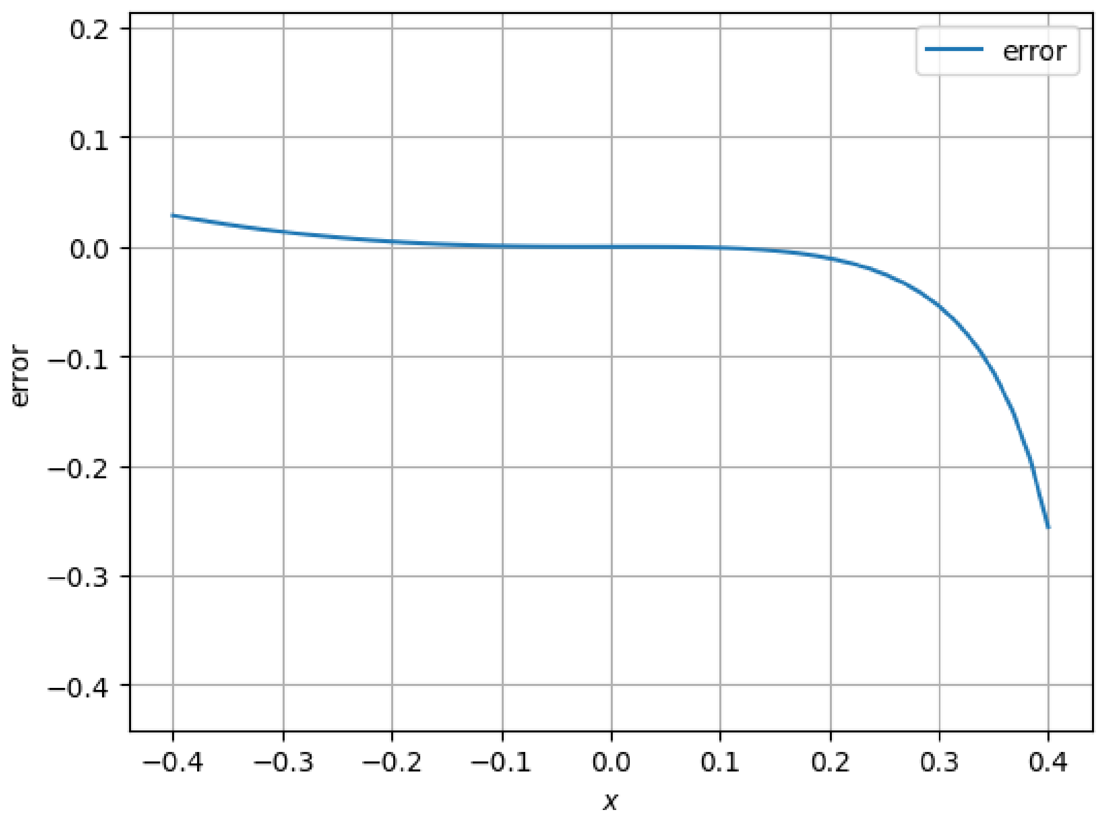
```
tn = 0
for i in range(n+1):
    tn += f.diff(x, i).subs(x, 0)/factorial(i) * x**i

plot_sympy(tn, pts, label="taylor")
plot_sympy(f, pts, label="f")
plt.legend(loc="best")
plt.axis("equal")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("function values")
```

```
Text(0, 0.5, 'function values')
```



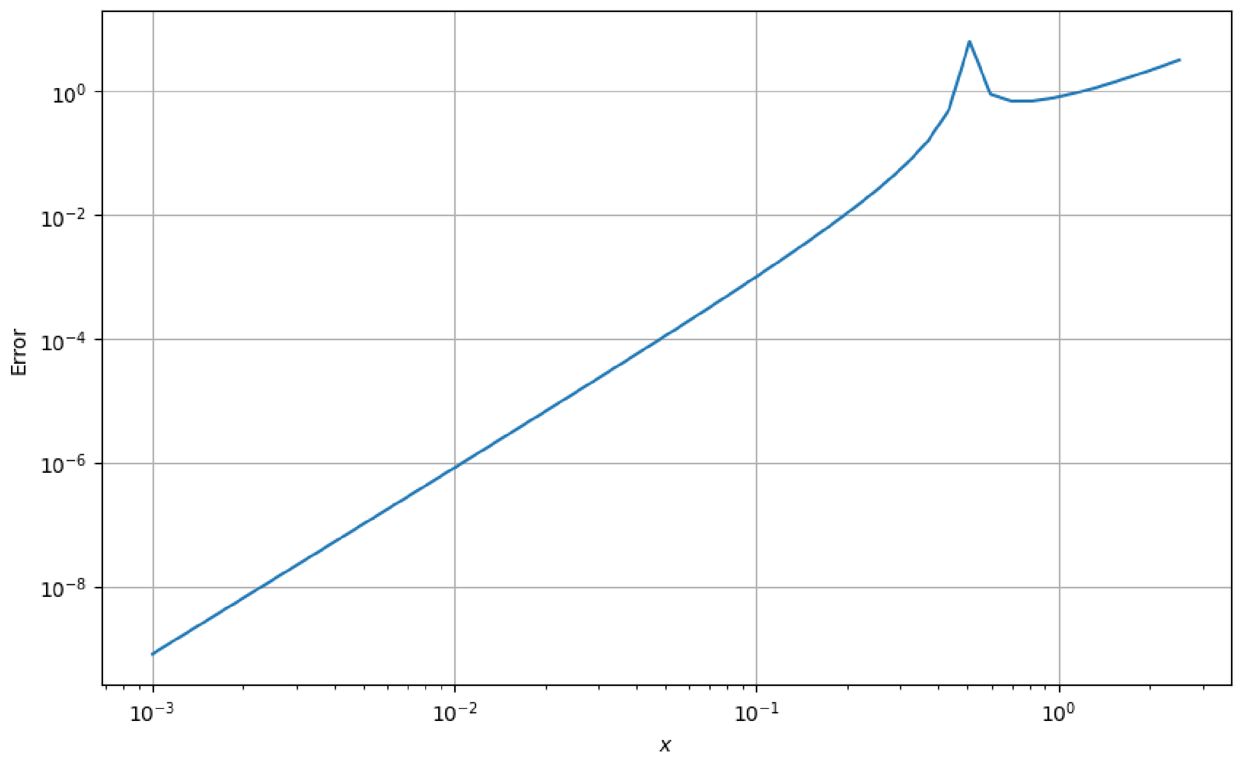
```
error = f - tn

plot_sympy(error, pts, label="error")
plt.legend(loc="best")
plt.ylim([-1.3, 1.3])
plt.axis("equal")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("error")
```

```
Text(0, 0.5, 'error')
```



```
# plot only points close to zero [10^(-3), 10^(0.4)]
plt.figure(figsize=(10, 6))
pos_pts = 10**np.linspace(-3, 0.4)
err_values = [abs(error.subs(x, pt)) for pt in pos_pts]
plt.loglog(pos_pts, err_values)
plt.grid()
plt.xlabel("$x$")
plt.ylabel("Error")
```

```
Text(0, 0.5, 'Error')
```



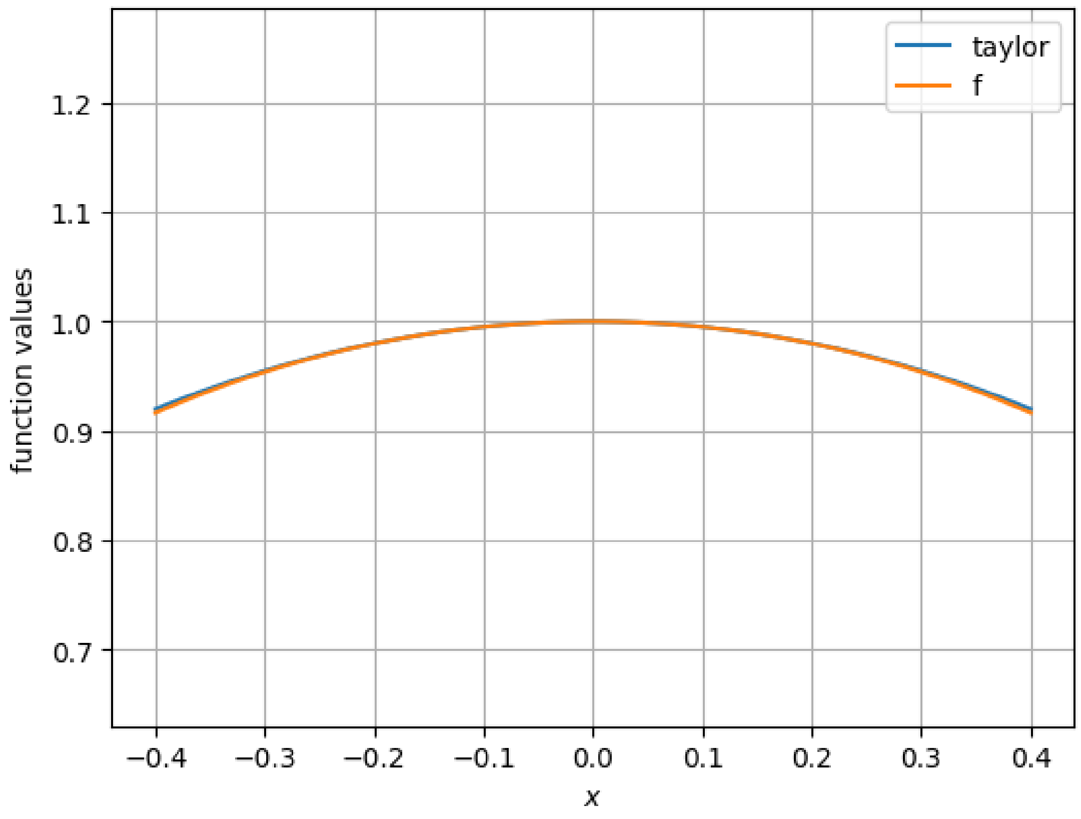
```
f = sp.sqrt(1-x**2)
f
```

$\sqrt{1-x^2}$

```
n = 2

tn = 0
for i in range(n+1):
    tn += f.diff(x, i).subs(x, 0)/factorial(i) * x**i

tn
```

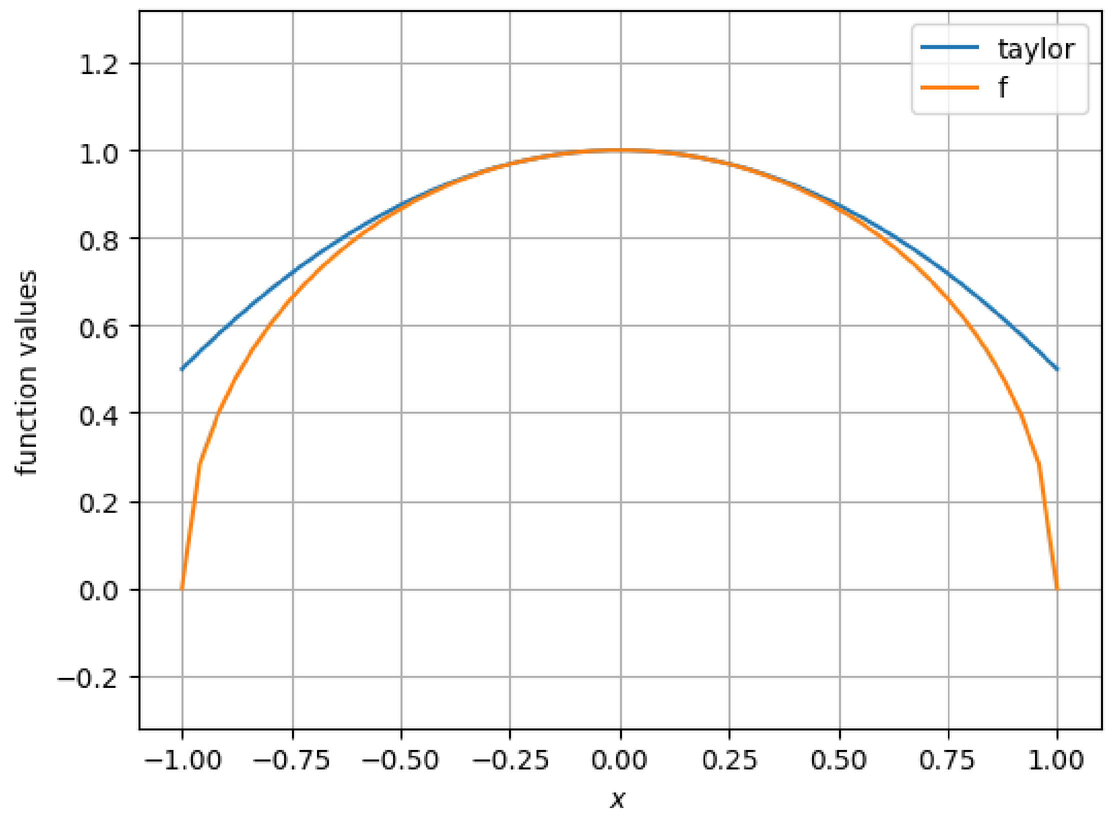
$1 - x^2/2$

```
plot_sympy(tn, pts, label="taylor")
plot_sympy(f, pts, label="f")
plt.legend(loc="best")
plt.ylim([-1.3, 1.3])
plt.axis("equal")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("function values")

Text(0, 0.5, 'function values')
```



```
pts = np.linspace(-1, 1)
plot_sympy(tn, pts, label="taylor")
plot_sympy(f, pts, label="f")
plt.legend(loc="best")
plt.ylim([-1.3, 1.3])
plt.axis("equal")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("function values")
```

```
Text(0, 0.5, 'function values')
```



```
f = sp.sqrt(x-10)
f
```

$\sqrt{x-10}$

```
n = 3
x0 = 12

tn = 0
for i in range(n+1):
    tn += f.diff(x, i).subs(x, x0)/factorial(i) * (x-x0)**i
tn
```

$\sqrt{2}(x-12)^3/128 - \sqrt{2}(x-12)^2/32 + \sqrt{2}(x-12)/4 + \sqrt{2}$

```
error1 = f.subs(x, 12.5) - tn.subs(x, 12.5).evalf()
abs(error1)
```

```
0.000183952061507453
```

```
error2 = f.subs(x, 12.25) - tn.subs(x, 12.25).evalf()
abs(error2)
```

$1.24076489040892 \cdot 10^{-5}$

```
f = sp.exp(x)
f
```

e^x

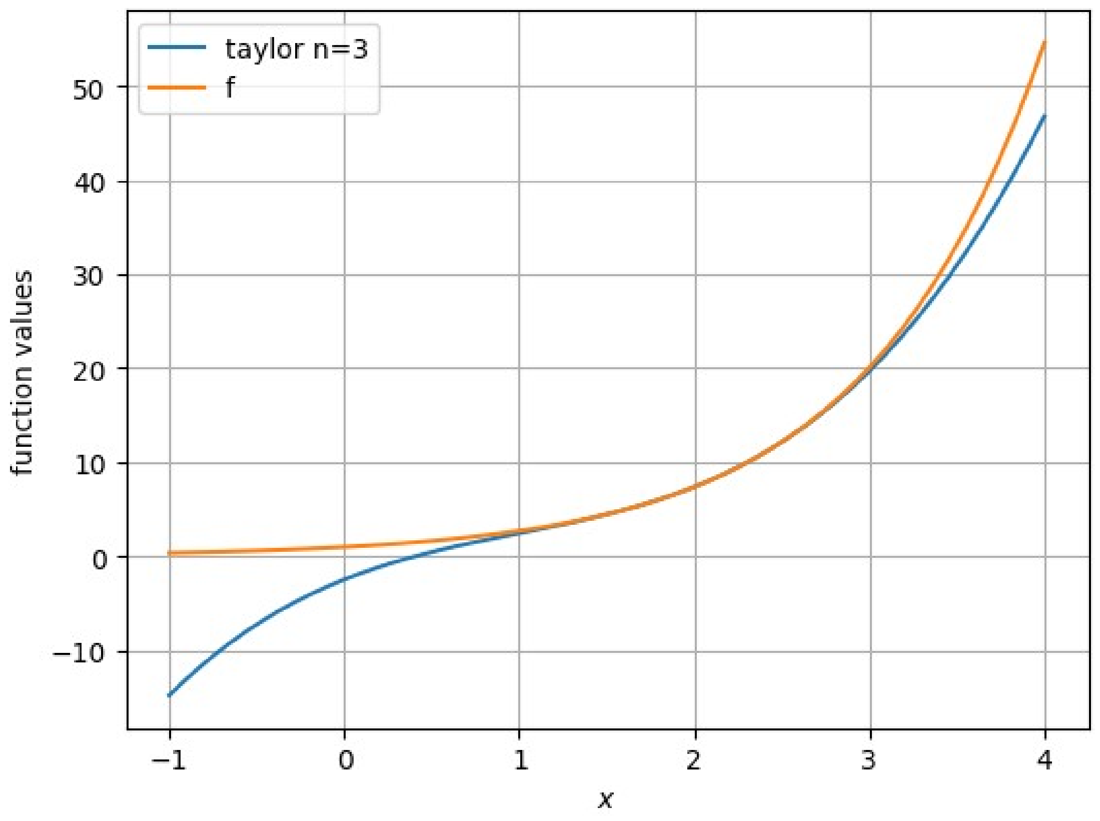
```
f.diff(x, 4)
```

e^x

```
def plot_sympy(my_f, my_pts, **kwargs):
    f_values = np.array([my_f.subs(x, pt) for pt in my_pts])
    plt.plot(pts, f_values, **kwargs)

def semilogy_sympy(my_f, my_pts, **kwargs):
    f_values = np.array([my_f.subs(x, pt) for pt in my_pts])
    plt.semilogy(pts, f_values, **kwargs)

n = 3
xo = 2

taylor = 0
for i in range(n+1):
    taylor += f.diff(x, i).subs(x, xo)/factorial(i) * (x-xo)**i

error = f - taylor

taylor
```

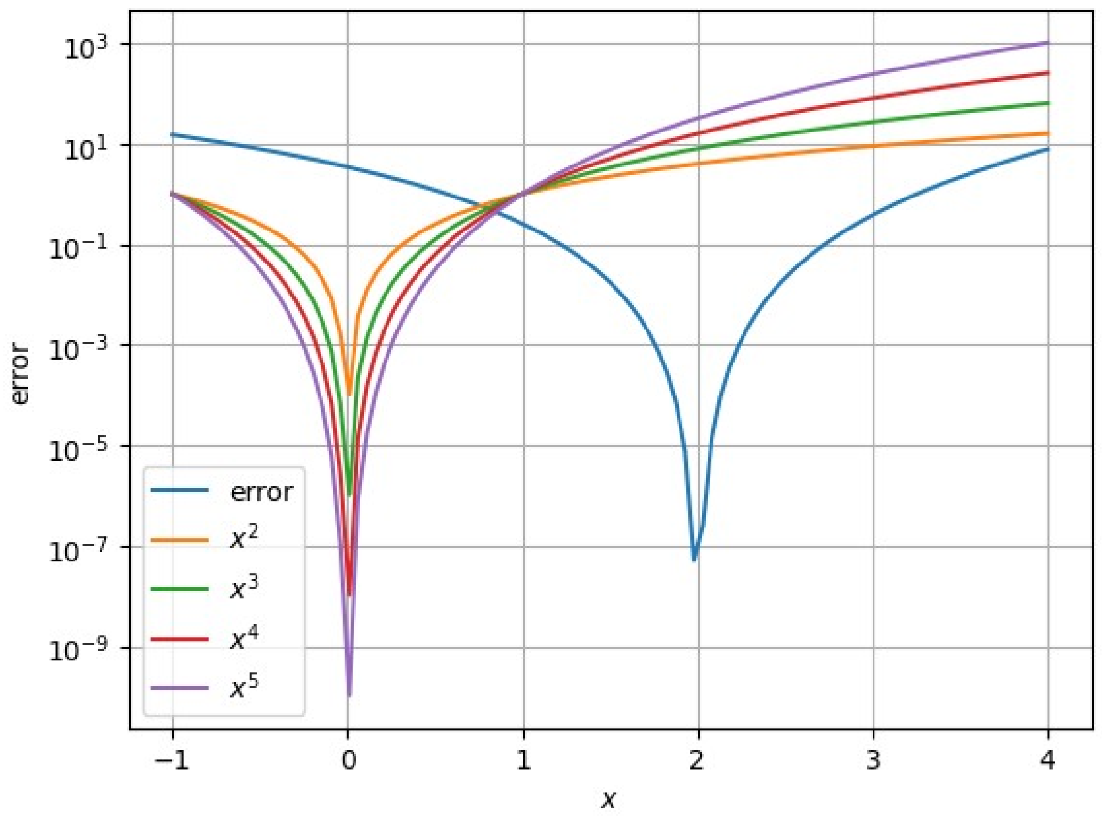
$(x-2)^3e^2/6 + (x-2)^2e^2/2 + (x-2)e^2 + e^2$

```
pts = np.linspace(-1, 4, 100)
plot_sympy(taylor, pts, label="taylor n=3")
plot_sympy(f, pts, label="f")
plt.legend(loc="best")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("function values")

Text(0, 0.5, 'function values')
```



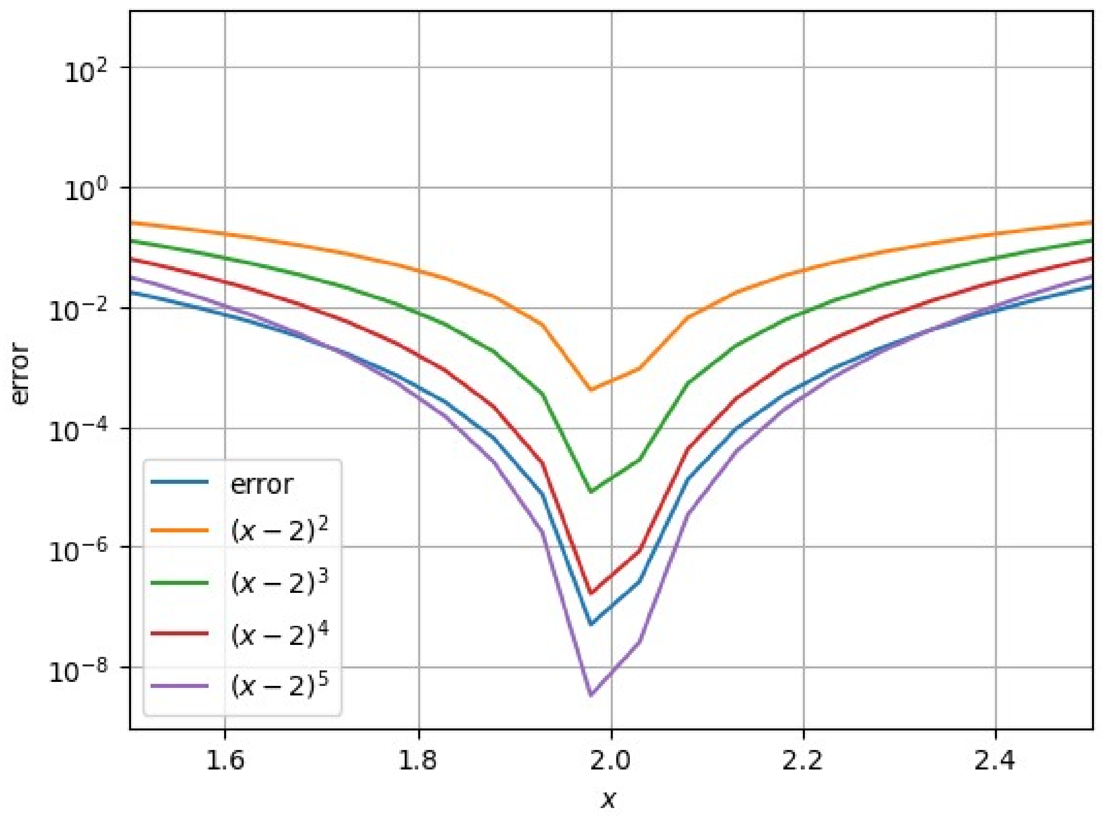
```
semilogy_sympy(error, pts, label="error")
f2=x**2
f3=x**3
f4=x**4
f5=x**5

semilogy_sympy(f2, pts, label="$x^2$")
semilogy_sympy(f3, pts, label="$x^3$")
semilogy_sympy(f4, pts, label="$x^4$")
semilogy_sympy(f5, pts, label="$x^5$")

plt.legend(loc="best")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("error")
```

```
Text(0, 0.5, 'error')
```



```
semilogy_sympy(error, pts, label="error")
f2=abs((x-2)**2)
f3=abs((x-2)**3)
f4=abs((x-2)**4)
f5=abs((x-2)**5)

semilogy_sympy(f2, pts, label="$(x-2)^2$")
semilogy_sympy(f3, pts, label="$(x-2)^3$")
semilogy_sympy(f4, pts, label="$(x-2)^4$")
semilogy_sympy(f5, pts, label="$(x-2)^5$")

plt.legend(loc="best")
plt.grid()
plt.xlabel("$x$")
plt.ylabel("error")
plt.xlim([1.5, 2.5])
```

```
(1.5, 2.5)
```



error
$(x-2)^2$
$(x-2)^3$
$(x-2)^4$
$(x-2)^5$
10^2
10^0
10^{-2}
10^{-4}
10^{-6}
10^{-8}
1.6
1.8
2.0
2.2
2.4
x