ロバストな回帰直線 least sum abs

青木繁伸

1　目的

観察値と予測値の差の絶対値の p 乗の和を最小にするという方法による回帰直線のパラメータを推定する。
p が 2 のときは，普通の最小二乗法である。
p が 1 のときは，絶対値の和を最小とするものである。

2　使用法

```
from least_sum_abs import least_sum_abs
least_sum_abs(x, y, bootstrap=1, a=1, sig=0.95, verbose=True)
```

2.1　引数

x	変数ベクトル
y	変数ベクトル
bootstrap	2 以上の値を指定することにより，ブートストラップ法によるパラメータ（切片と傾き）の信頼限界を求める（少なくとも 1000 以上が望ましい）
p	1 のとき絶対値の和を最小にする（2 のときは普通の最小二乗法）
sig	信頼率（デフォルトは 0.95）
verbose	必要最小限のプリント出力をする

2.2　戻り値の名前

"intercept"	切片
"slope"	傾き
"intercept CL"	bootstlap を 2 以上にしたときに，切片の信頼限界
"slope CL"	bootstlap を 2 以上にしたときに，傾きの信頼限界

3　使用例

```
x = [2,1,3,2,4,3,5,6,5,4,1,4,5,6]
y = [2,3,2,4,5,4,6,6,7,5,6,6,5,6]

import sys
sys.path.append("statlib")
from least_sum_abs import least_sum_abs
```

```
a = least_sum_abs(x, y)
```

```
Intercept: 2.9992880872288605
    Slope: 0.5001198665578978
```

```
b = least_sum_abs(x, y, bootstrap=1000)
```

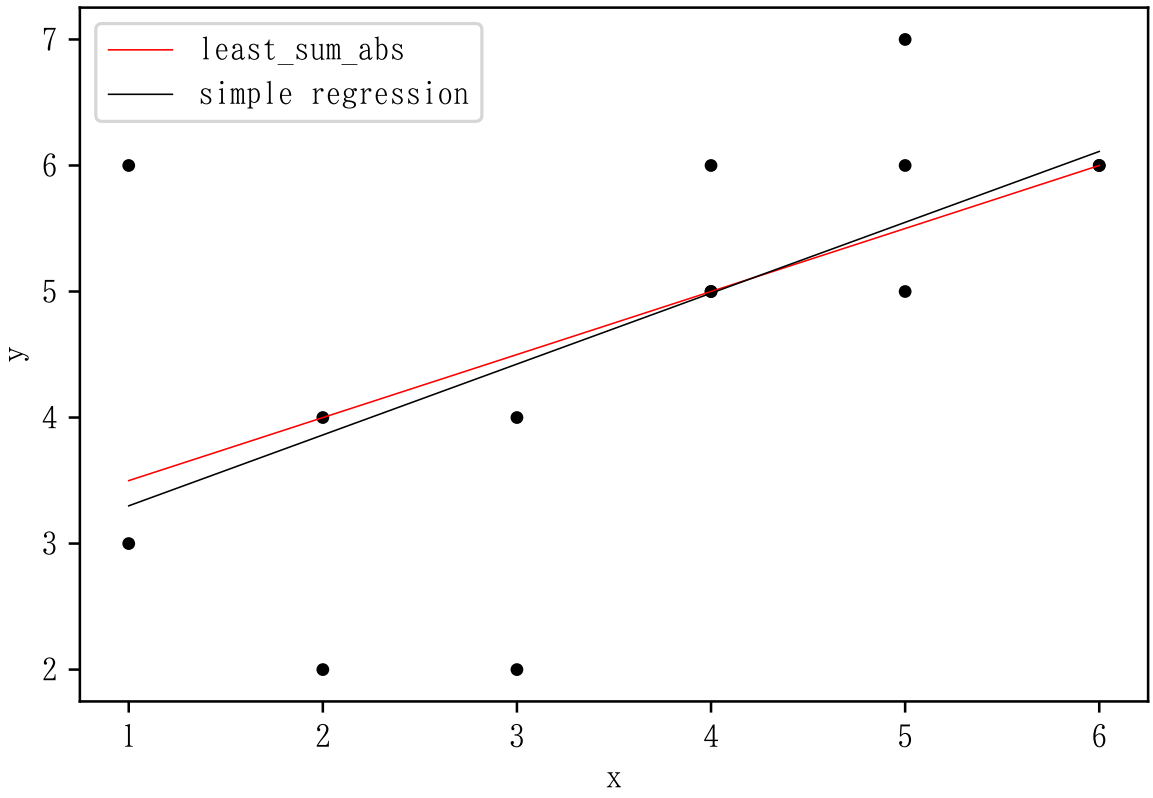
```
           Estimate  Lower C.L.  Upper C.L.
Intercept  2.999288    0.000056    5.999971
Slope      0.500120    0.000004    1.194447
```

```
import scipy as sp

def simple_reg(x, y):
    mx = sp.mean(x)
    my = sp.mean(y)
    slope = sum((x-mx)*(y-my))/sum((x-mx)**2)
    intercept = my-slope*mx
    return intercept, slope
```

```
import matplotlib.pyplot as plt

intercept = a["intercept"]
slope = a["slope"]
x0 = sp.amin(x)
x1 = sp.amax(x)
x2 = sp.array([x0, x1])
y2 = intercept + slope * x2
intercept_reg, slope_reg = simple_reg(x, y)
y3 = intercept_reg + slope_reg * x2
plt.scatter(x, y, c="black", s=9)
plt.plot(x2, y2, label="least_sum_abs", linewidth=0.5, color="red")
plt.plot(x2, y3, label="simple regression", linewidth=0.5, color="
   black")
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.show()
```


least_sum_abs
simple regression
7
6
5
4
3
2
y
1
2
3
4
5
6
x