PLS 回帰

青木繁伸

2020 年 3 月 17 日

1　目的

5 種類の方法により PLS 回帰を行う。

R の **pls** パッケージに含まれる cppls()，kernelpls()，widekernelpls()，simpls()，oscorespls() を **Python** に翻訳・修正したものである。

R の **pls** の情報

```
Package:            pls
Title:              Partial Least Squares and Principal Component Regression
Version:            2.7-0
Date:               2018-08-20
Authors@R:          c(person("Bjørn-Helge", "Mevik", role = c("aut", "cre"), email =
                    "b-h@mevik.net"), person("Ron", "Wehrens", role = "aut"), person("Kristian
                    Hovde", "Liland", role = "aut"), person("Paul", "Hiemstra", role = "ctb"))
Author:             Bjørn-Helge Mevik [aut, cre], Ron Wehrens [aut], Kristian Hovde Liland [aut],
                    Paul Hiemstra [ctb]
Maintainer:         Bjørn-Helge Mevik <b-h@mevik.net>
Encoding:           UTF-8
LazyData:           yes
Description:        Multivariate regression methods Partial Least Squares Regression (PLSR),
                    Principal Component Regression (PCR) and Canonical Powered Partial Least
                    Squares (CPPLS).
Depends:            R (>= 2.10)
Imports:            grDevices, graphics, methods, stats
Suggests:           MASS, parallel, Rmpi, testthat, RUnit
License:            GPL-2
URL:                http://mevik.net/work/software/pls.html, https://github.com/bhmevik/pls
BugReports:         https://github.com/bhmevik/pls/issues
NeedsCompilation:   no
Packaged:           2018-08-20 18:36:26 UTC; bhm
Repository:         CRAN
Date/Publication:   2018-08-21 05:10:11 UTC
```

```
Built:              R 3.5.0; ; 2018-08-22 14:03:28 UTC; unix
```

参考文献

cppls	Indahl, U. (2005) A twist to partial least squares regression. *Journal of Chemometrics*, **19**, 32 – 44.
	Liland, K.H and Indahl, U.G (2009) Powered partial least squares discriminant analysis, *Journal of Chemometrics*, **23**, 7 – 18.
	Indahl, U.G., Liland, K.H. and Næs, T. (2009) Canonical partial least squares - a unified PLS approach to classification and regression problems. *Journal of Chemometrics*, **23**, 495 – 504.
kernelpls	de Jong, S. and ter Braak, C. J. F. (1994) Comments on the PLS kernel algorithm. *Journal of Chemometrics*, **8**, 169 – 174.
	Dayal, B. S. and MacGregor, J. F. (1997) Improved PLS algorithms. *Journal of Chemometrics*, **11**, 73 – 85.
widekernelpls	Rännar, S., Lindgren, F., Geladi, P. and Wold, S. (1994) A PLS Kernel Algorithm for Data Sets with Many Variables and Fewer Objects. Part 1: Theory and Algorithm. *Journal of Chemometrics*, **8**, 111 – 125.
simpls	de Jong, S. (1993) SIMPLS: an alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, **18**, 251 – 263.
oscorespls	Martens, H., Næs, T. (1989) *Multivariate calibration*. Chichester: Wiley.

2 使用法

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

from multi import cppls
cppls(x, y, ncomp, yAdd=None, center=True, lower=0.5, upper=0.5, truncPow=False,
     weights=None, verbose=True)

from multi import kernelpls
kernelpls(x, y, ncomp=None, center=True, verbose=True)

from multi import widekernelpls
widekernelpls(x, y, ncomp=None, center=True,
     tol=2.22044604925031e-16 ** 0.5, maxit=200, verbose=True)

from multi import simpls
simpls(x, y, ncomp=None, center=True, verbose=True)

from multi import oscorespls
oscorespls(x, y, ncomp=None, center=True, tol=2.22044604925031e-16 ** 0.5,
```

```
     maxit=200, verbose=True)
```

予測値，スコアのプロット

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

from multi import pls_plot
pls_plot(obj, type="p", ncomp=None, ny=1, ax1=1, ax2=2,
     txt=None, color="blue", color2= "red", alpha=0.3)
```

新しいデータに対する予測値を求める

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

from multi import pls_predict
pls_predict(obj, newX=None, ncomp=None, comps=None, type="response", verbose=True):
```

2.1 引数

x	独立変数データ行列
y	従属変数データ行列（2 列以上でもよい）
ncomp	使用する主成分の個数（指定しなくてもよい）
yAdd	付加的な従属変数データ行列
center	デフォルトで各変数を中心化する True
lower	パワー最適化のパラメータの下限値（デフォルトは 0.5）
upper	パワー最適化のパラメータの上限値（デフォルトは 0.5）
truncPow	True の場合は実験的なパワーアルゴリズムを使う（デフォルトは False）
weights	各観測値の重み（デフォルトでは重み付けしない None）
verbose	必要最小限のプリント出力をする
obj	pls 回帰プログラムの戻り値
type	実測値と予測値の対比プロットの場合（デフォルト）は "p"，スコアの二次元配置をプロットする場合は "s" を指定する
ncomp	回帰に使うコンポーネント数（デフォルトは最大数）
ny	複数の従属変数を分析した場合は何番目の従属変数かを表す（デフォルトは 1）
ax1	横軸にとるスコアの番号（デフォルトは 1）
ax2	縦軸にとるスコアの番号（デフォルトは 2）
color	ドットの描画色（デフォルトは青）
txt	ドットに添えるテキスト（デフォルトは None）
color2	ドットに添えるテキストの描画色（デフォルトは赤）
alpha	アルファチャネル（デフォルトは 0.3）
newX	新しいデータセット（独立変数データ行列のみ） デフォルト（None）の場合は分析に用いたデータセットが仮定される

ncomp	使用する合成変数の数。表示されるのはそれぞれの合成変数に対して。
comps	使用する合成変数のセット〔デフォルトの場合は None)。表示されるのは総合した 1 個のモデルにおける値。
type	type="response" の場合（デフォルト）は予測値を返す。type="score" の場合は comps で指定された合成変数の値を返す。

2.2 戻り値の名前

"coefficients"	ncomp までの主成分に対する回帰係数
"scores"	独立変数のスコア
"loadings"	独立変数の負荷量
"loadingWeights"	独立変数の負荷量の重み
"yScores"	従属変数のスコア
"yLoadings"	従属変数の負荷量
"projection"	独立変数データ行列をスコアに変換するための射影
"xMeans"	独立変数データの平均値ベクトル
"yMeans"	従属変数データの平均値ベクトル
"fittedValues"	予測値
"residuals"	残差
"xVariances"	各主成分で説明される独立変数の分散
"xTotalVariance"	独立変数の全分散（xVariances の合計）
"gammas"	γ 値
"cancors"	正準相関係数
"A"	正準相関係数の重みベクトル
"smallNorm"	0 か 0 に近い説明変数の位置

3 使用例

3.1 3 変数を使って 1 変数を予測する例

```
import pandas as pd

df = pd.read_csv("../Python/data/iris.csv")
x = df.loc[:, ["sw", "pl", "pw"]]
y = pd.DataFrame(df.loc[:, "sl"])

import sys
sys.path.append("statlib")
from multi import cppls
a = cppls(x, y, ncomp=3)
```

```
***** Coefficients
```

```
1 comp
          sl
sw -0.011534
pl  0.346379
pw  0.140331

2 comps
          sl
sw  0.613899
pl  0.450242
pw  0.050192

3 comps
          sl
sw  0.650837
pl  0.709132
pw -0.556483

***** Scores

           Comp1     Comp2     Comp3
Obj1    2.573133 -0.190991 -0.018997
Obj2    2.557709  0.298453  0.009857
Obj3    2.656517  0.118932  0.038761
Obj4    2.468155  0.184308 -0.036359
Obj5    2.576218 -0.288879 -0.024767
...          ...       ...       ...
Obj146 -1.750709 -0.023007  0.463283
Obj147 -1.430731  0.442517  0.193912
Obj148 -1.638116 -0.065331  0.178947
Obj149 -1.923647 -0.447074  0.359309
Obj150 -1.470415 -0.077290  0.029835

[150 rows x 3 columns]

***** Loadings

       Comp1     Comp2     Comp3
sw  0.097012 -1.029812  0.191703
pl -0.919635 -0.007579 -0.363036
pw -0.386534 -0.065935  0.911841
```

```
                     Comp1     Comp2     Comp3
SS loadings       1.004549  1.064917  1.000000
Prop. var.        0.334850  0.354972  0.333333
Cumu. prop. var.  0.334850  0.689822  1.023156

***** Loading weiths

       Comp1     Comp2     Comp3
sw  0.030848 -0.980968  0.191703
pl -0.926385 -0.100077 -0.363036
pw -0.375312  0.166392  0.911841

                     Comp1     Comp2     Comp3
SS loadings       1.000000  1.000000  1.000000
Prop. var.        0.333333  0.333333  0.333333
Cumu. prop. var.  0.333333  0.666667  1.000000

***** y scores

            Comp1     Comp2     Comp3
Obj1   -13.639855 -9.297016 -9.260971
Obj2   -13.104959 -9.464407 -9.456908
Obj3   -12.570063 -9.029682 -9.051884
Obj4   -12.302615 -8.828314 -8.775766
Obj5   -13.372407 -9.044419 -9.003067
...           ...       ...       ...
Obj146 -17.919026 -9.438857 -9.884831
Obj147 -16.849233 -9.465582 -9.642136
Obj148 -17.384129 -9.149398 -9.311566
Obj149 -16.581785 -8.131018 -8.475417
Obj150 -15.779440 -8.296498 -8.311119

[150 rows x 3 columns]

***** y loadings

       Comp1     Comp2     Comp3
sl -0.373904 -0.638923 -0.640097

                     Comp1     Comp2     Comp3
SS loadings       0.139804  0.408223  0.409724
```

```
Prop. var.         0.139804  0.408223  0.409724
Cumu. prop. var.   0.139804  0.548027  0.957751

Projection

       Comp1     Comp2     Comp3
sw  0.030848 -0.978887 -0.057707
pl -0.926385 -0.162559 -0.404454
pw -0.375312  0.141078  0.947787

***** Fitted values

1 comp
              sl
Obj1    4.881228
Obj2    4.886995
Obj3    4.850050
Obj4    4.920479
Obj5    4.880074
...          ...
Obj146  6.497931
Obj147  6.378290
Obj148  6.455832
Obj149  6.562593
Obj150  6.393128

[150 rows x 1 columns]

2 comps
              sl
Obj1    5.003256
Obj2    4.696306
Obj3    4.774062
Obj4    4.802721
Obj5    5.064646
...          ...
Obj146  6.512631
Obj147  6.095556
Obj148  6.497573
Obj149  6.848239
Obj150  6.442510
```

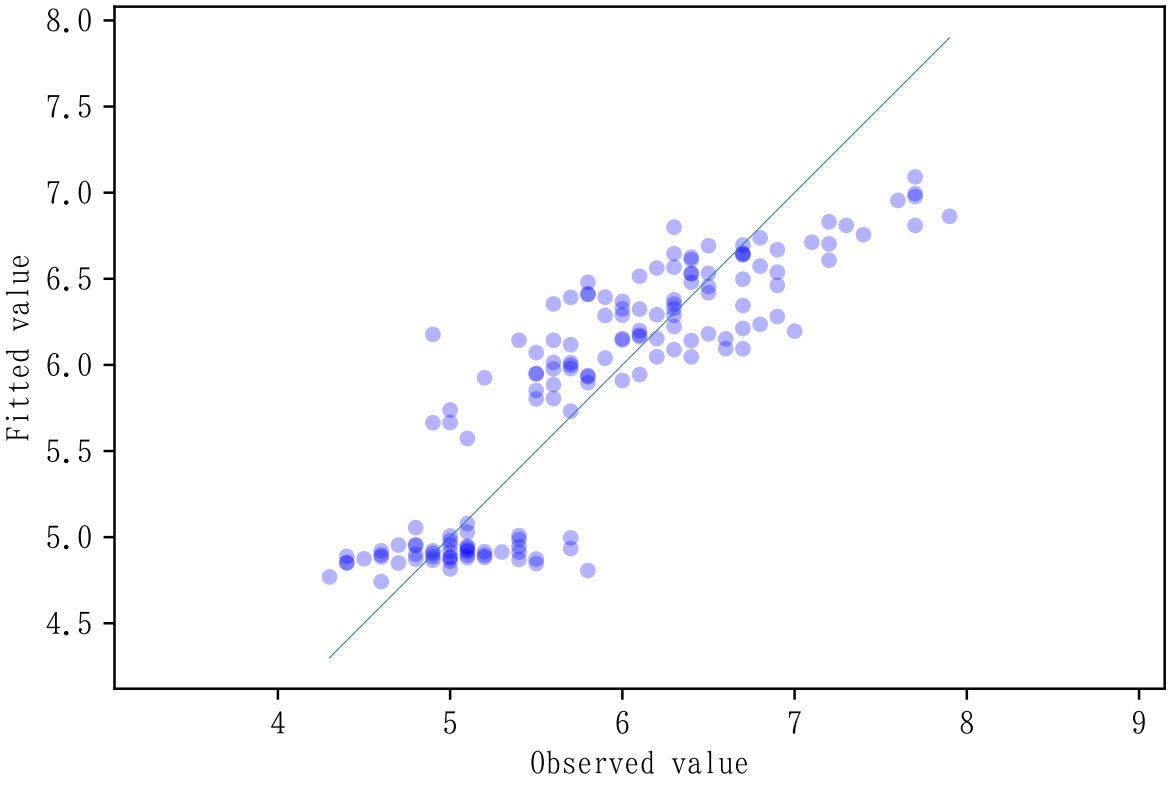
```
[150 rows x 1 columns]

3 comps
              sl
Obj1    5.015416
Obj2    4.689997
Obj3    4.749251
Obj4    4.825994
Obj5    5.080499
...          ...
Obj146  6.216085
Obj147  5.971433
Obj148  6.383030
Obj149  6.618246
Obj150  6.423413

[150 rows x 1 columns]

***** Residuals

1 comp
              sl
Obj1    0.218772
Obj2    0.013005
Obj3   -0.150050
Obj4   -0.320479
Obj5    0.119926
...          ...
Obj146  0.202069
Obj147 -0.078290
Obj148  0.044168
Obj149 -0.362593
Obj150 -0.493128

[150 rows x 1 columns]

2 comps
              sl
Obj1    0.096744
Obj2    0.203694
Obj3   -0.074062
Obj4   -0.202721
```

```
Obj5   -0.064646
...          ...
Obj146  0.187369
Obj147  0.204444
Obj148  0.002427
Obj149 -0.648239
Obj150 -0.542510

[150 rows x 1 columns]

3 comps
              sl
Obj1    0.084584
Obj2    0.210003
Obj3   -0.049251
Obj4   -0.225994
Obj5   -0.080499
...          ...
Obj146  0.483915
Obj147  0.328567
Obj148  0.116970
Obj149 -0.418246
Obj150 -0.523413

[150 rows x 1 columns]

***** x means
sw  3.057333
pl  3.758000
pw  1.199333

***** y means
sl  5.843333

***** Explained variances of x by each component
Comp1  550.658541
Comp2   23.041092
Comp3    5.502634

***** Total variance of x
579.20227
```

```
***** Gammas
[0.5 0.5 0.5]

***** Canonical correlations
[0.01467038 0.0016908  0.00043159]

***** A
[[-0.0025437  -0.18982961 -1.47737782]]
```

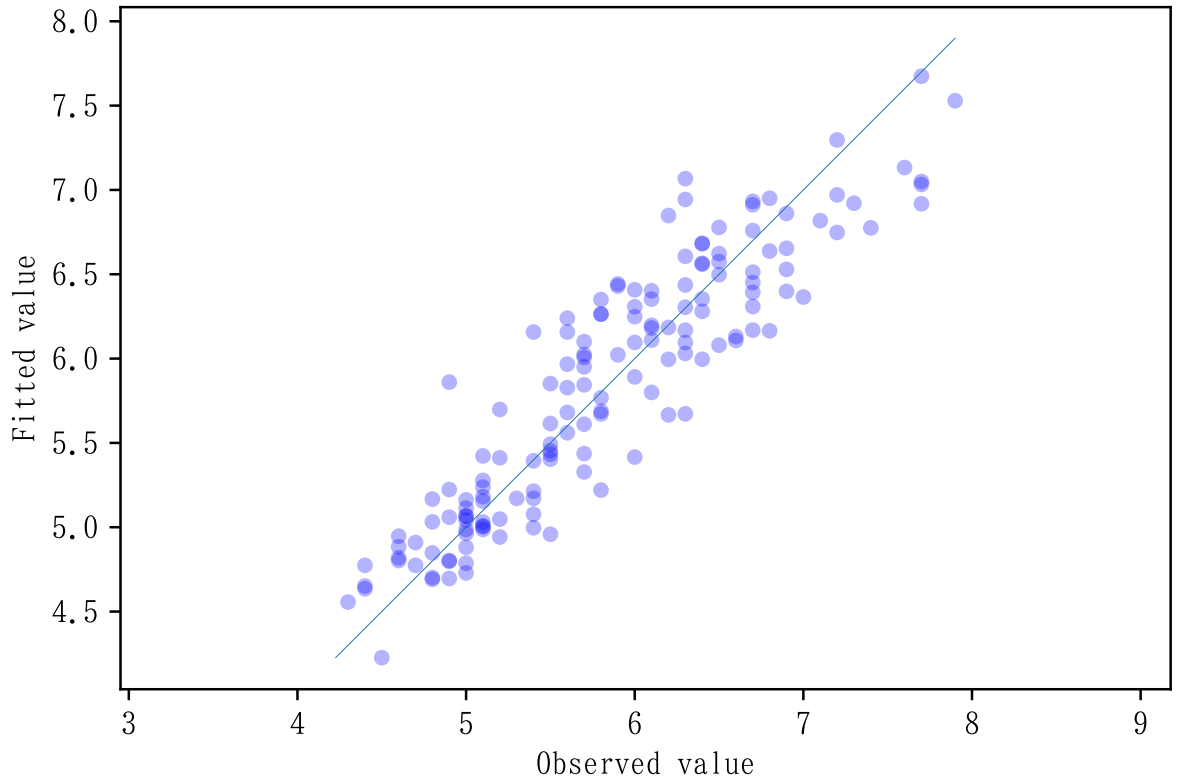
実測値と予測値の関係図
合成変数 1 個を使う場合

```
import sys
sys.path.append("statlib")
from multi import pls_plot

pls_plot(a, ncomp=1)
```



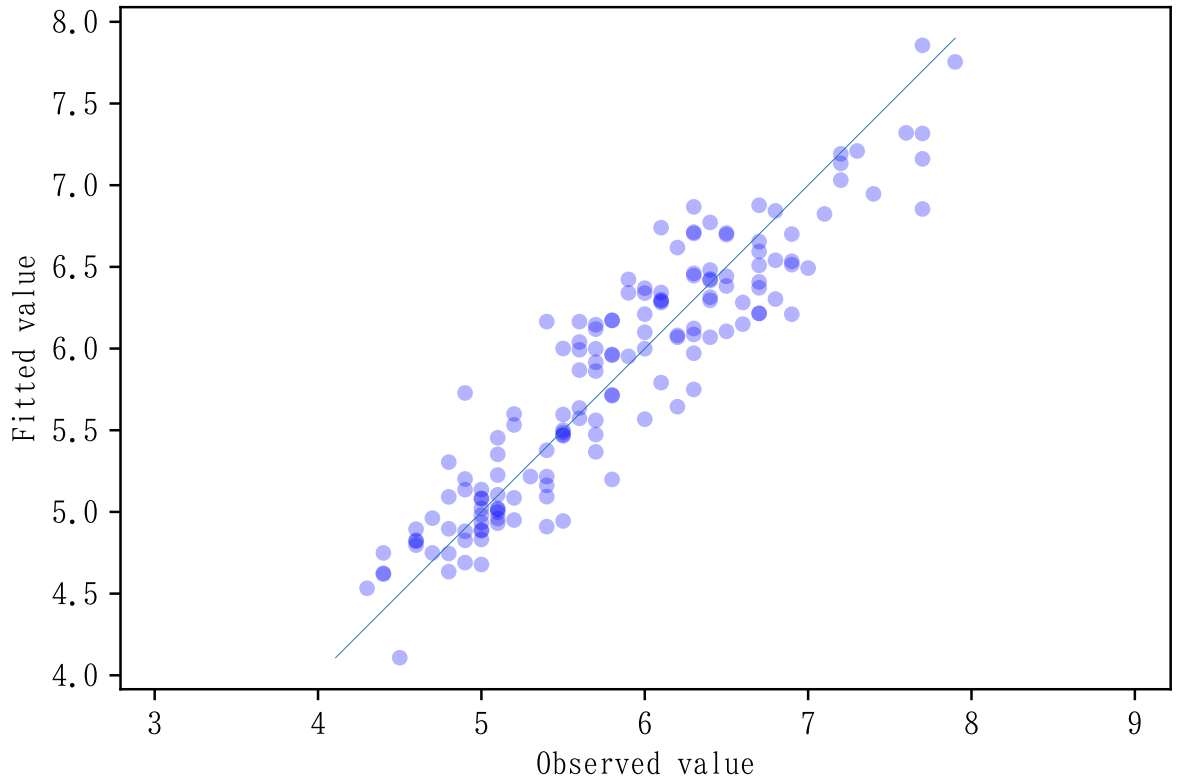
合成変数 2 個を使う場合

```
pls_plot(a, ncomp=2)
```

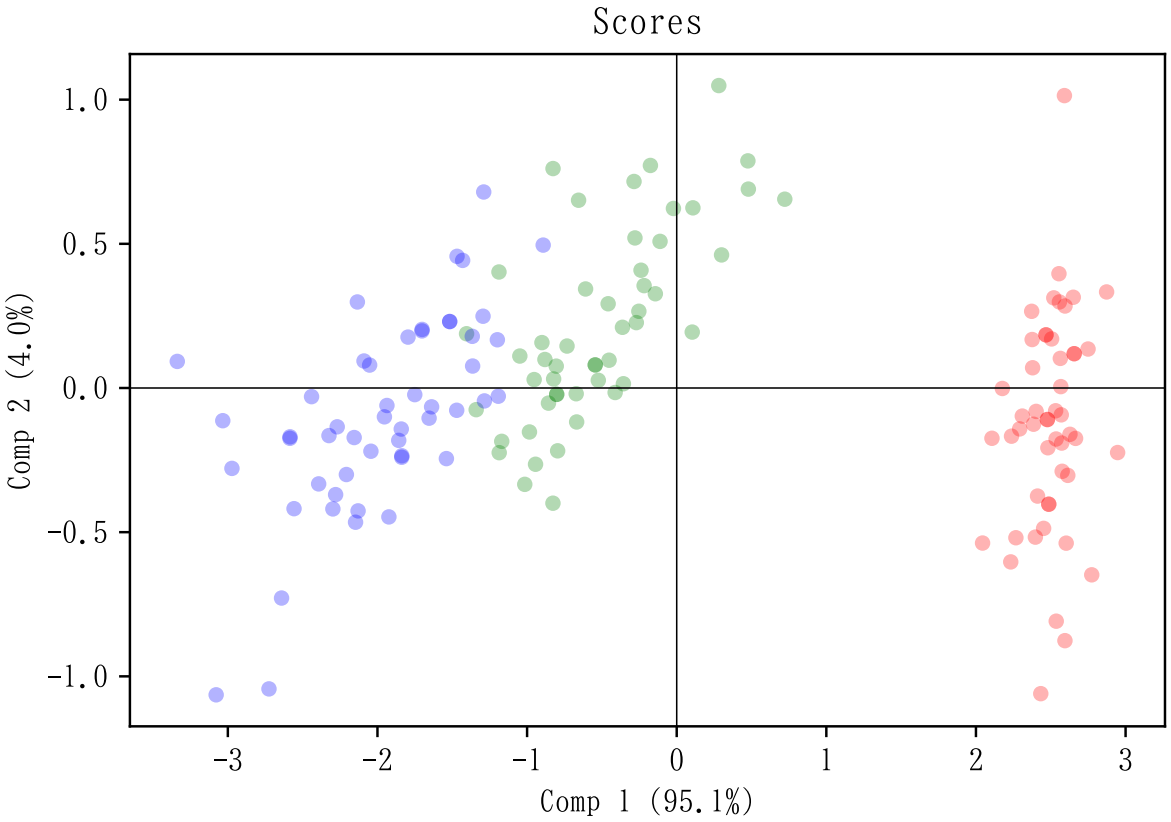


合成変数 3 個を使う場合

```
pls_plot(a, ncomp=3)
```




```
import numpy as np
color = np.hstack((np.repeat("red", 50), np.repeat("green", 50), np.
    repeat("blue", 50)))
pls_plot(a, type="s", ncomp=[1, 2], color=color)
```



3.2　2 変数を使って 2 変数を予測する例

```
df = pd.read_csv("../Python/data/iris.csv")
x = df.iloc[:, :2]
y = df.iloc[:, 2:4]
a = cppls(x, y, ncomp=2)
```

```
***** Coefficients

1 comp
          pl        pw
sl  1.863112  0.747580
sw -0.866620 -0.347734

2 comps
          pl        pw
sl  1.775593  0.723292
sw -1.338623 -0.478721

***** Scores
```

```
           Comp1     Comp2
Obj1    0.860684 -0.309506
Obj2    0.831150  0.235805
Obj3    1.096843  0.070394
Obj4    1.145338  0.190752
Obj5    0.993531 -0.392212
...          ...       ...
Obj146 -0.800929 -0.103066
Obj147 -0.649122  0.479898
Obj148 -0.619587 -0.065414
Obj149 -0.178873 -0.415063
Obj150 -0.075561  0.047543

[150 rows x 2 columns]

***** Loadings

       Comp1     Comp2
sl -1.015318 -0.421753
sw  0.188262 -0.906711

                     Comp1     Comp2
SS loadings       1.066314  1.000000
Prop. var.        0.533157  0.500000
Cumu. prop. var.  0.533157  1.033157

***** Loading weithts

       Comp1     Comp2
sl -0.906711 -0.421753
sw  0.421753 -0.906711

                  Comp1  Comp2
SS loadings         1.0    1.0
Prop. var.          0.5    0.5
Cumu. prop. var.    0.5    1.0

***** y scores

           Comp1      Comp2
Obj1   -0.620485   7.142100
```

```
Obj2   -0.620485   6.462082
Obj3   -0.578567   7.639587
Obj4   -0.662402   8.139871
Obj5   -0.620485   7.830716
...          ...        ...
Obj146 -2.566563   8.110645
Obj147 -2.415449   7.598913
Obj148 -2.516104   8.733811
Obj149 -2.650398  11.659285
Obj150 -2.440547  10.791569

[150 rows x 2 columns]

***** y loadings

       Comp1     Comp2
pl -2.054803  0.464882
pw -0.824497  0.129011

                     Comp1     Comp2
SS loadings       4.902012  0.232759
Prop. var.        2.451006  0.116380
Cumu. prop. var.  2.451006  2.567386

Projection

       Comp1     Comp2
sl -0.906711 -0.188262
sw  0.421753 -1.015318

***** Fitted values

1 comp
              pl        pw
Obj1    1.989463  0.489702
Obj2    2.050150  0.514053
Obj3    1.504204  0.294990
Obj4    1.404555  0.255006
Obj5    1.716490  0.380171
...          ...       ...
Obj146  5.403752  1.859697
Obj147  5.091817  1.734532
```

```
Obj148  5.031130  1.710181
Obj149  4.125548  1.346813
Obj150  3.913263  1.261633

[150 rows x 2 columns]

2 comps
              pl        pw
Obj1    1.845579  0.449772
Obj2    2.159772  0.544475
Obj3    1.536929  0.304072
Obj4    1.493232  0.279615
Obj5    1.534157  0.329571
...          ...       ...
Obj146  5.355839  1.846400
Obj147  5.314913  1.796444
Obj148  5.000720  1.701742
Obj149  3.932593  1.293266
Obj150  3.935365  1.267767

[150 rows x 2 columns]

***** Residuals

1 comp
              pl        pw
Obj1   -0.589463 -0.289702
Obj2   -0.650150 -0.314053
Obj3   -0.204204 -0.094990
Obj4    0.095445 -0.055006
Obj5   -0.316490 -0.180171
...          ...       ...
Obj146 -0.203752  0.440303
Obj147 -0.091817  0.165468
Obj148  0.168870  0.289819
Obj149  1.274452  0.953187
Obj150  1.186737  0.538367

[150 rows x 2 columns]

2 comps
              pl        pw
```

```
Obj1   -0.445579 -0.249772
Obj2   -0.759772 -0.344475
Obj3   -0.236929 -0.104072
Obj4    0.006768 -0.079615
Obj5   -0.134157 -0.129571
...          ...       ...
Obj146 -0.155839  0.453600
Obj147 -0.314913  0.103556
Obj148  0.199280  0.298258
Obj149  1.467407  1.006734
Obj150  1.164635  0.532233

[150 rows x 2 columns]

***** x means
sl  5.843333
sw  3.057333

***** y means
pl  3.758000
pw  1.199333

***** Explained variances of x by each component
Comp1  100.090457
Comp2   30.384809

***** Total variance of x
130.47527

***** Gammas
[0.5 0.5]

***** Canonical correlations
[0.32184625 0.00485945]

***** A
[[-0.05983197  0.15677097]
 [ 0.13283316  0.        ]]
```

3.3 新しいデータセットに対する予測

cppls() による分析結果を obj に代入する。

```
import pandas as pd

df = pd.read_csv("../Python/data/pls.csv")
x = df.loc[:, ["X1", "X2", "X3", "X4", "X5"]]
y = df.loc[:, ["Y1", "Y2", "Y3"]]

import sys
sys.path.append("statlib")
from multi import cppls, pls_predict
obj = cppls(x, y, verbose=False)
```

新しいデータセット（$n = 4$）

```
newX = [[71.5, 39.5, 71.6, 46.1, 41.9],
        [38.4, 40.9, 47.2, 50.9, 53.9],
        [49.8, 54.2, 41.7, 27.0, 72.8],
        [53.0, 42.3, 71.9, 64.7, 65.2]]
```

ncomp を省略すると，理論的に可能な合成変数の数までのそれぞれについて予測値を求める。

この場合は合成変数を 1 個だけ，2 個まで，...，5 個までに対して予測値を求める。1 〜 3 列は，Y1，Y2，Y3 の 3 変数に対する予測値である。行数 4 は $n = 4$ に対応する。4 行列が comp=5 個表示される。

```
a = pls_predict(obj, newX=newX)
```

```
***** Predicted values
[[[53.93437133 52.42366065 73.25389697]
  [48.84498667 49.28798941 36.71220461]
  [46.32095657 47.7328842  18.58971283]
  [50.56297354 50.34647533 49.04732009]]

 [[58.33638332 51.93717508 74.24251925]
  [49.63206424 49.20100603 36.88896979]
  [52.36319954 47.06512954 19.94670506]
  [55.40388662 49.81148489 50.13451262]]

 [[57.42496753 51.11311248 76.46305091]
  [49.47792779 49.06164254 37.26450078]
  [52.35734184 47.05983326 19.96097649]
  [55.81056829 50.17918881 49.14369202]]

 [[60.16128321 53.12409669 78.652381  ]
  [46.00883968 46.51212623 34.48887842]
  [55.84530954 49.62322463 22.75170444]
  [59.0248579  52.54144757 51.71544973]]
```

```
 [[59.81989093 52.70381925 78.30231213]
  [47.55825186 48.41955949 36.07766941]
  [53.11102888 46.25713657 19.94793116]
  [64.74794061 59.58695707 57.58398641]]]
```

4 個までの合成変数を使った場合（ncomp=4）の予測値を求める。

```
a = pls_predict(obj, newX=newX, ncomp=4)
```

```
***** Predicted values
[[[60.16128321 53.12409669 78.652381  ]
  [46.00883968 46.51212623 34.48887842]
  [55.84530954 49.62322463 22.75170444]
  [59.0248579  52.54144757 51.71544973]]]
```

1 個だけの合成変数を使った場合，2 個までの合成変数を使った場合（ncomp=4）の予測値を求める。

```
a = pls_predict(obj, newX=newX, ncomp=[1,2])
```

```
***** Predicted values
[[[53.93437133 52.42366065 73.25389697]
  [48.84498667 49.28798941 36.71220461]
  [46.32095657 47.7328842  18.58971283]
  [50.56297354 50.34647533 49.04732009]]

 [[58.33638332 51.93717508 74.24251925]
  [49.63206424 49.20100603 36.88896979]
  [52.36319954 47.06512954 19.94670506]
  [55.40388662 49.81148489 50.13451262]]]
```

1，2，3，4 番目の合成変数を使った場合の予測値を求める。これは ncomp=4 を指定したときと同じ結果になる。

```
a = pls_predict(obj, newX=newX, comps=[1,2,3,4])
```

```
***** Predicted values
[[60.16128321 53.12409669 78.652381  ]
 [46.00883968 46.51212623 34.48887842]
 [55.84530954 49.62322463 22.75170444]
 [59.0248579  52.54144757 51.71544973]]
```

1，4 番目の合成変数を使った場合の予測値を求める。

```
a = pls_predict(obj, newX=newX, comps=[1,4])
```

```
***** Predicted values
```

```
[[56.67068701 54.43464486 75.44322706]
 [45.37589856 46.7384731  33.93658225]
 [49.80892427 50.29627556 21.38044077]
 [53.77726314 52.70873408 51.6190778 ]]
```

これ以降はスコア（合成変数の値）を求める（type="score"）。
5 個の合成変数の値を求める。

```
a = pls_predict(obj, newX=newX, type="score")
```

```
***** Predicted scores
[[-25.6925055   11.40795704 -12.91503197   8.74765423  -1.47354701]
 [  7.53410319   2.03973708  -2.18415926 -11.09023478   6.6877074 ]
 [ 24.01251194  15.65866891  -0.08300537  11.15059042 -11.80193958]
 [ -3.6819634   12.54538347   5.76279995  10.27567626  24.70246647]]
```

4 番目の合成変数の値を求める。

```
a = pls_predict(obj, newX=newX, ncomp=4, type="score")
```

```
***** Predicted scores
[[  8.74765423]
 [-11.09023478]
 [ 11.15059042]
 [ 10.27567626]]
```

1 番目，2 番目の合成変数の値を求める。

```
a = pls_predict(obj, newX=newX, ncomp=[1,2], type="score")
```

```
***** Predicted scores
[[-25.6925055   11.40795704]
 [  7.53410319   2.03973708]
 [ 24.01251194  15.65866891]
 [ -3.6819634   12.54538347]]
```

1 番目，2 番目，3 番目，4 番目の合成変数の値を求める。

```
a = pls_predict(obj, newX=newX, comps=[1,2,3,4], type="score")
```

```
***** Predicted scores
[[-25.6925055   11.40795704 -12.91503197   8.74765423]
 [  7.53410319   2.03973708  -2.18415926 -11.09023478]
 [ 24.01251194  15.65866891  -0.08300537  11.15059042]
 [ -3.6819634   12.54538347   5.76279995  10.27567626]]
```

1 番目，4 番目の合成変数の値を求める。

```
a = pls_predict(obj, newX=newX, comps=[1,4], type="score")
```

```
***** Predicted scores
[[-25.6925055    8.74765423]
 [  7.53410319 -11.09023478]
 [ 24.01251194  11.15059042]
 [ -3.6819634   10.27567626]]
```