ASCA

This is a short guide for the ASCA module in python.

Please see README for description of parameters, attributes and methods.

This guide is made with simulated data:
A 900 × 500 Y matrix containing response variables was created based on a mean, two predictor variables and a random error term:

$$\boldsymbol{Y} = \boldsymbol{\mu} + \boldsymbol{\alpha} + \boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

All simulated data are based on a standard normal distribution.
The design of the predictor variables are a full factorial design based on the variables in table 1. The 3^2 full factorial design results in 9 combinations, which are repeated 100 times. The ASCA algorithm is called by:

Table 1: Two factors each with three levels.

George	Michael
Bush	Schumacher
Lucas	Bublé
Clooney	Jordan

```
mdl = ASCA(X, Y)
```

and the model is fit by:

```
mdl.fit()
```

The raw data can be plotted by the command:

```
mdl.plot_raw()
```

And coloured by specifying the factor, either by:

```
mdl.plot_raw(0)
```

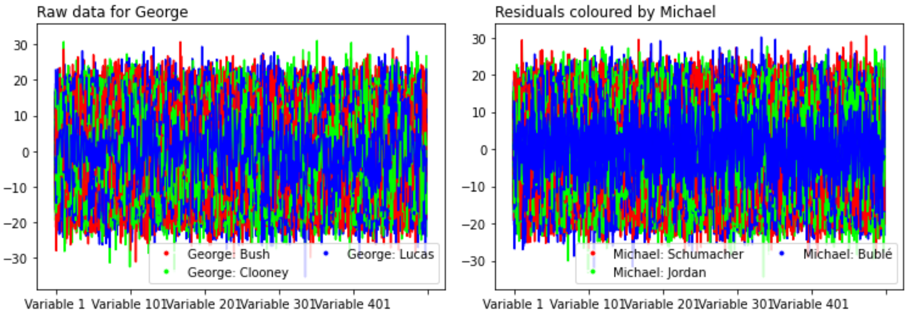
or:

```
mdl.plot_raw('George')
```

for the first factor. The result of the plot is depicted in figure 3. Similarly, the residuals can be plotted and coloured by the specific factor by:

```
mdl.plot_residual('Michael')
```

Which can be seen in figure 1b. The number of permutations can be set in options:



(a) Raw plot of the first factor

(b) residual plot of the second factor

Figure 1: Raw and residual plots

```
mdl.Options.n_perm = 10000
```

where 1000 is the default.
In `.Options` the number of components can be specified, along with whether to have interaction effects:

```
mdl.Options.n_components = 5
```

and

```
mdl.Options.interaction = True
```

Interactions are by default set to `False`.
A permutation test can be performed by:

```
mdl.permutation_test()
```

and the results can be visualised as a histogram with:

```
mdl.plot_permutations()
```

The result is depicted in figure 2.
Score plots can be plotted by

```
mdl.plot_scores()
```

which by default returns the scores of the first component. The specific scores can be plotted by

```
mdl.plot_scores(0)
```

or

```
mdl.plot_scores('George')
```

By default the score plots are coloured by the plotted factor, but they can also be coloured by a specific factor:

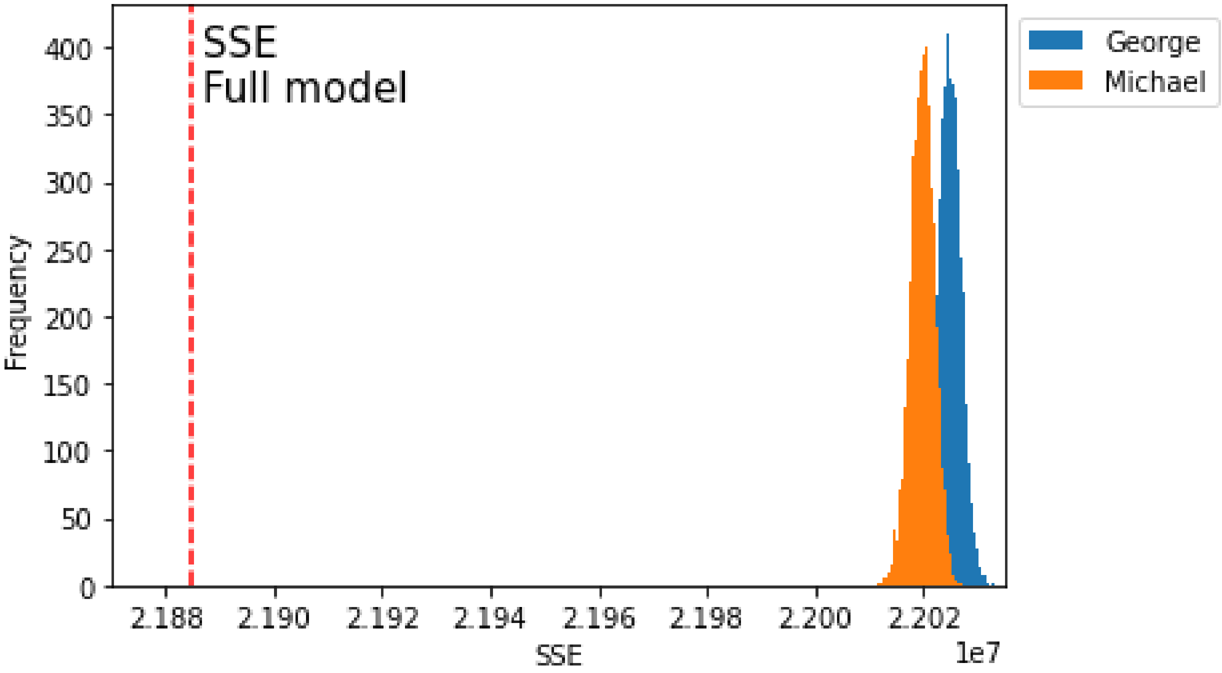


Figure 2: Histogram of the permutation test

```
mdl.plot_scores(factor = 'George', group_by = 'Michael')
```

The two resulting score plots are seen in figure 3a and 3b.
The score plots can also be coloured by a different vector, not specified in the model:

```
mdl.plot_scores(factor = 'George', group_by = Wham)
```

The component of interest can be specified by the command, component:

```
mdl.plot_scores(factor = 'George', group_by = Wham, component = (0,1))
```

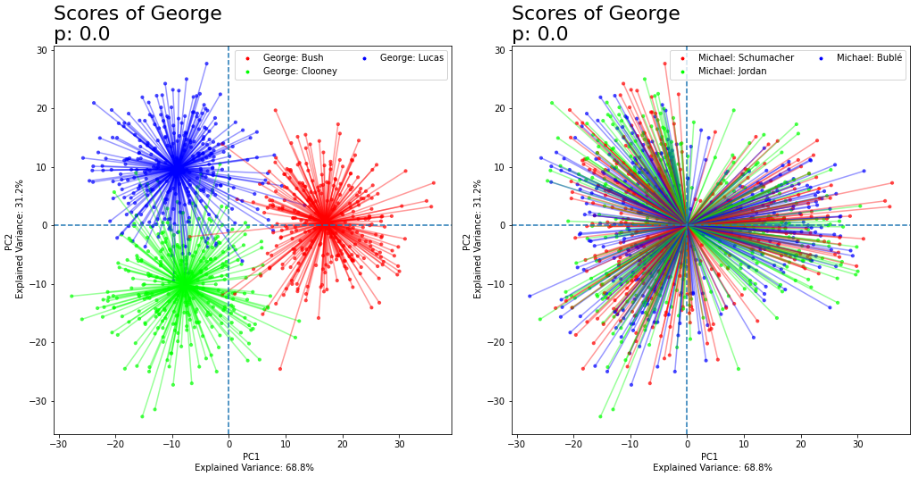
where the default is the first and second component, $(0, 1)$. The component is given by a tuple of two integers.
The loadings can be plotted by

```
mdl.plot_loadings()
```

where the default is determined by the first component. Similarly to the other plot types the factor can be specified by

```
mdl.plot_loadings(0)
```

or



(a) Coloured by the first factor 'George'

(b) Coloured by the second factor 'Michael'

Figure 3: Score plots of the first factor 'George'

```
mdl.plot_loadings('George')
```

The results of the plot can be seen in figure 4.

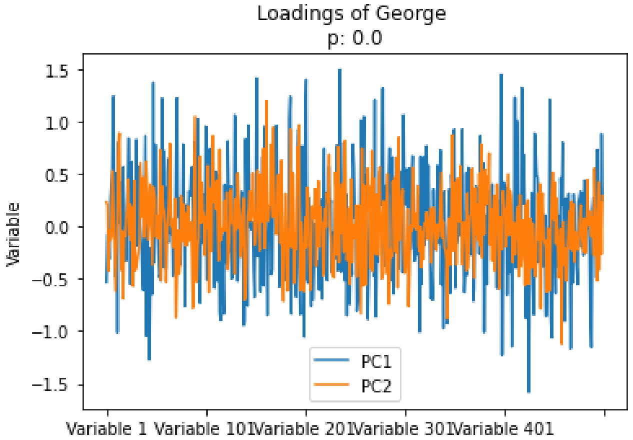


Figure 4: Loading plot of the first factor 'George'