cornerhex

Sebastian Stammler

Mar 09, 2023

CONTENTS:

1 Installation **3**

1.1 1. Quickstart 3

1.2 2. Customizations 5

1.3 Appendix A: Contributing/Bugs/Features 23

1.4 Module Reference 23

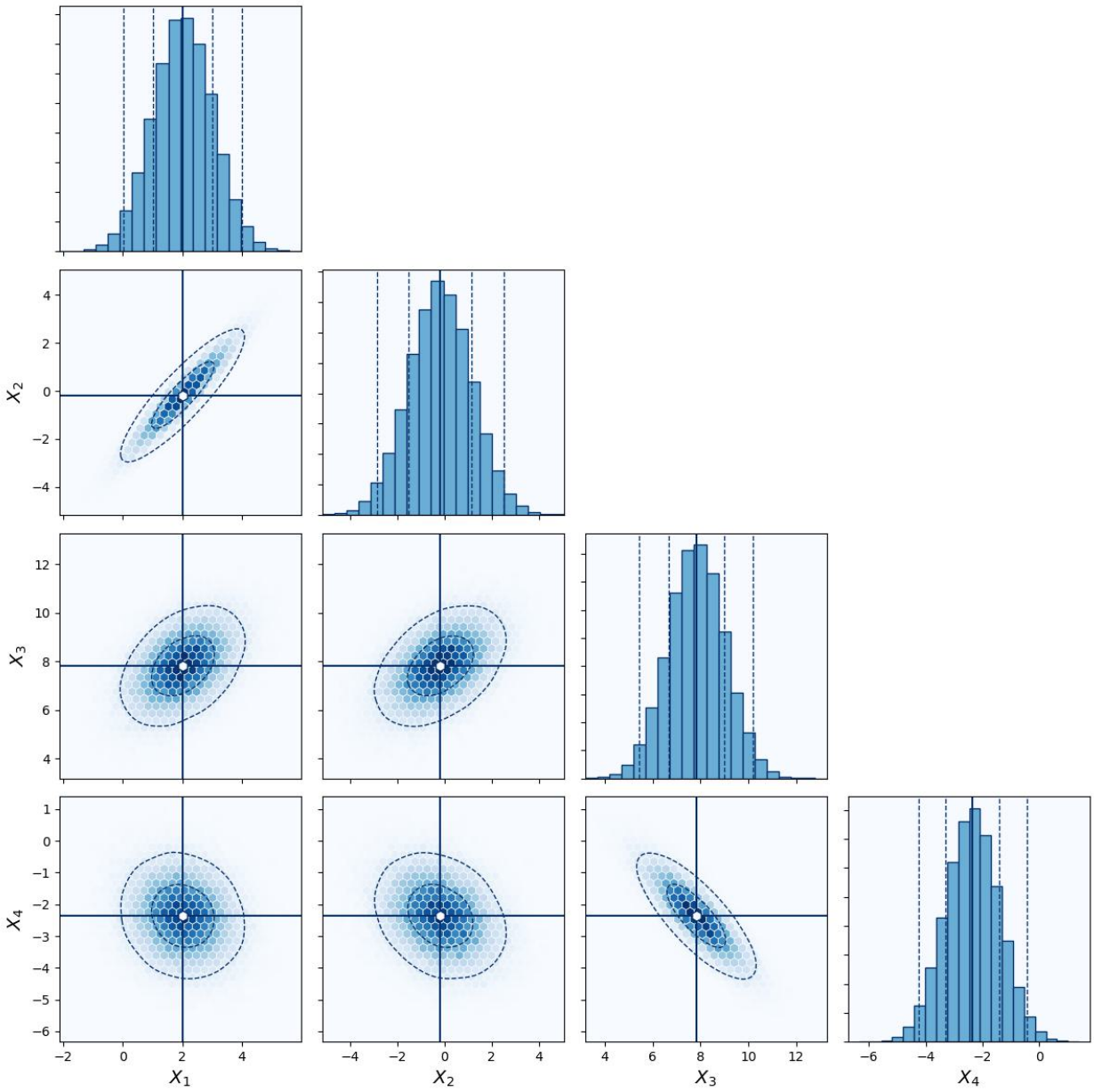
2 Indices and tables **25**

Python Module Index **27**

Index **29**

`cornerhex` is a package to visualize multidimensional data in matrix corner plots, for example the results of Markov Chain Monte Carlo (MCMC) methods.

Instead of 2d histograms or scatter plots it uses `matplotlib.pyplot.hexbin`. `cornerhex` can be easily costumized with different color schemes.



CHAPTER

ONE

INSTALLATION

To install `cornerhex` simply type
`pip install cornerhex`

Please have a look at the following examples to learn how to use `cornerhex`.

1.1 1. Quickstart

`cornerhex` can be used to easily visualize multidimensional data in a hexbin matrix plot, for example the results of Marcov Chain Monte Carlo (MCMC) methods.

1.1.1 Creating random data

We first need to create some random multi-dimensional data. For this purpose we create a random covariance matrix with 4 dimensions and pick 50000 random samples.

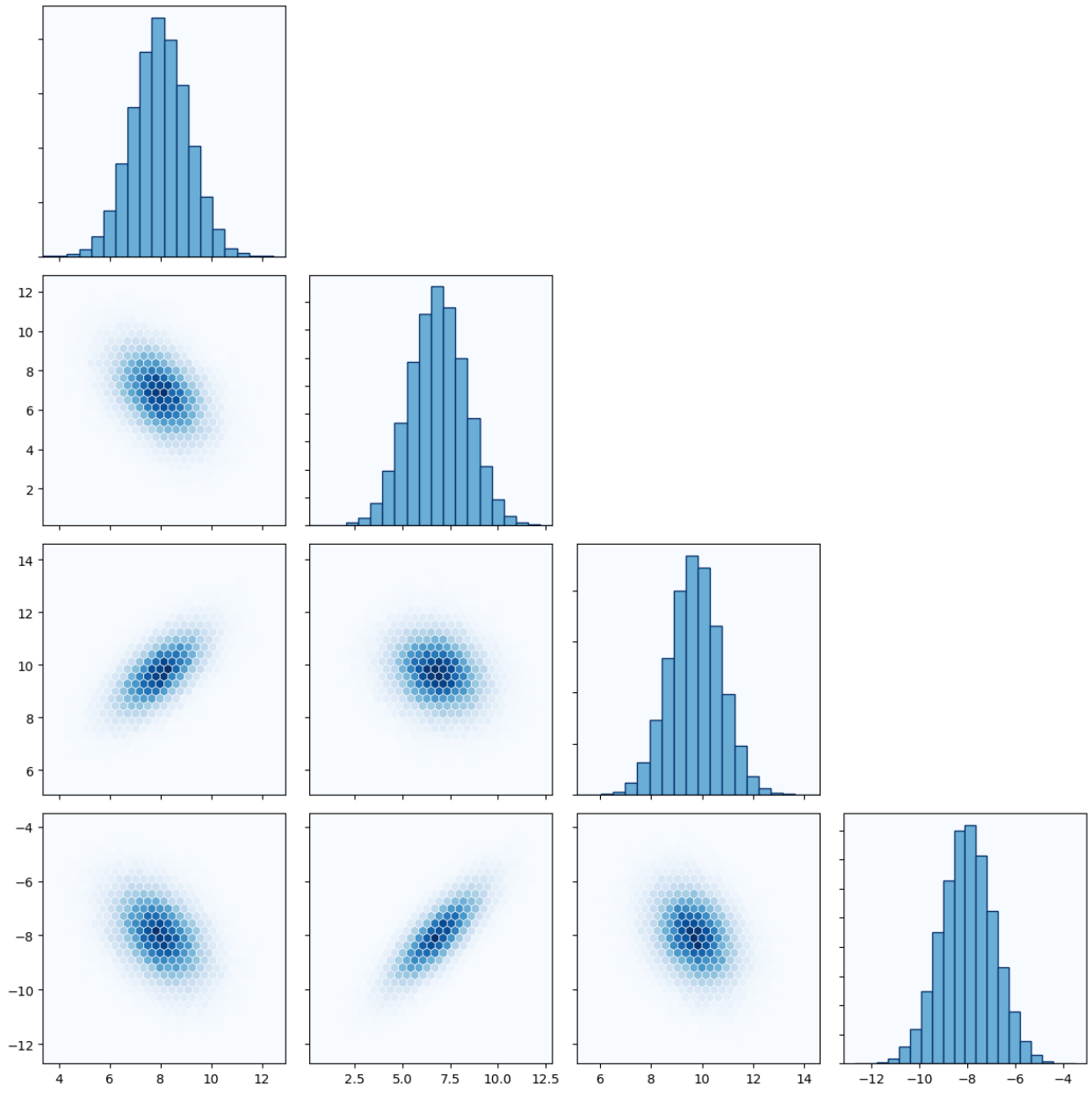
[1]:
```
Ndims, Nsamples = 4, 50_000
```

[2]:
```
import numpy as np

D = 2.*np.random.rand(Ndims, Ndims)-1.
cov = np.dot(D, D.T)
mean = 20.*np.random.rand(Ndims) - 10.

data = np.random.multivariate_normal(mean, cov, size=Nsamples)
```

1.1.2 Creating cornerhex plot

This data array can be directly fed into `cornerhex.cornerplot`.

[3]:
```
from cornerhex import cornerplot

cornerplot(data);
```



The next chapter discusses customizations of the `cornerplot`.

1.2 2. Customizations

This chapter discusses the various customizations that can be applied to `cornerhex.cornerplot`. For this puporse we create random data with 50000 samples and 4 features.

[1]:
```
Ndims, Nsamples = 4, 50_000
```

[2]:
```
import numpy as np

D = 2.*np.random.rand(Ndims, Ndims)-1.
cov = np.dot(D, D.T)
mean = 20.*np.random.rand(Ndims) - 10.

data = np.random.multivariate_normal(mean, cov, size=Nsamples)
```

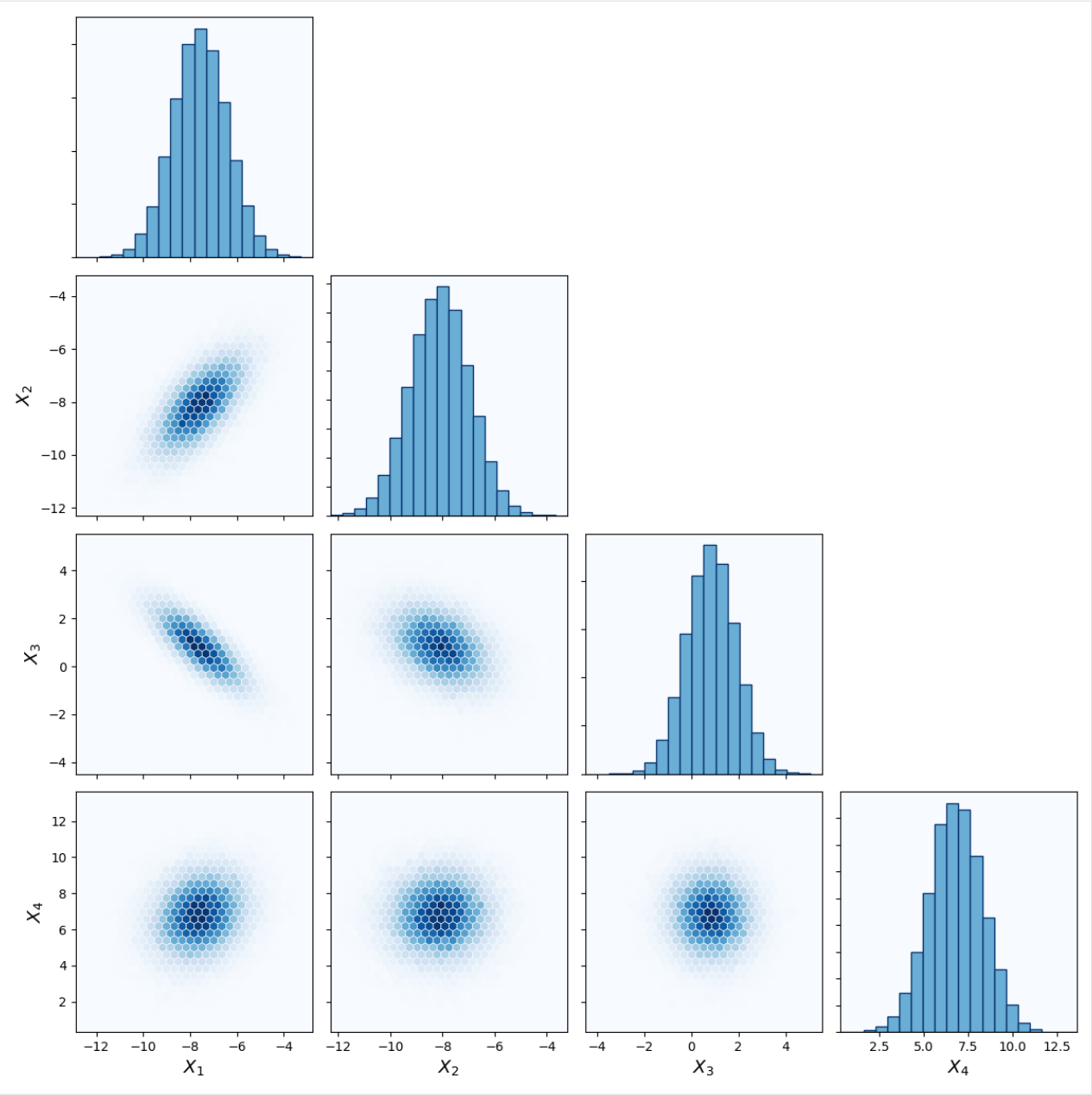
1.2.1 Adding feature labels

It is possible to add labels of the data features. LaTeX notion is allowed. We simply call the features X_i here.

[3]:
```
labels = ["$X_{{{}}}$".format(i+1) for i in range(Ndims)]
```

[4]:
```
from cornerhex import cornerplot
```

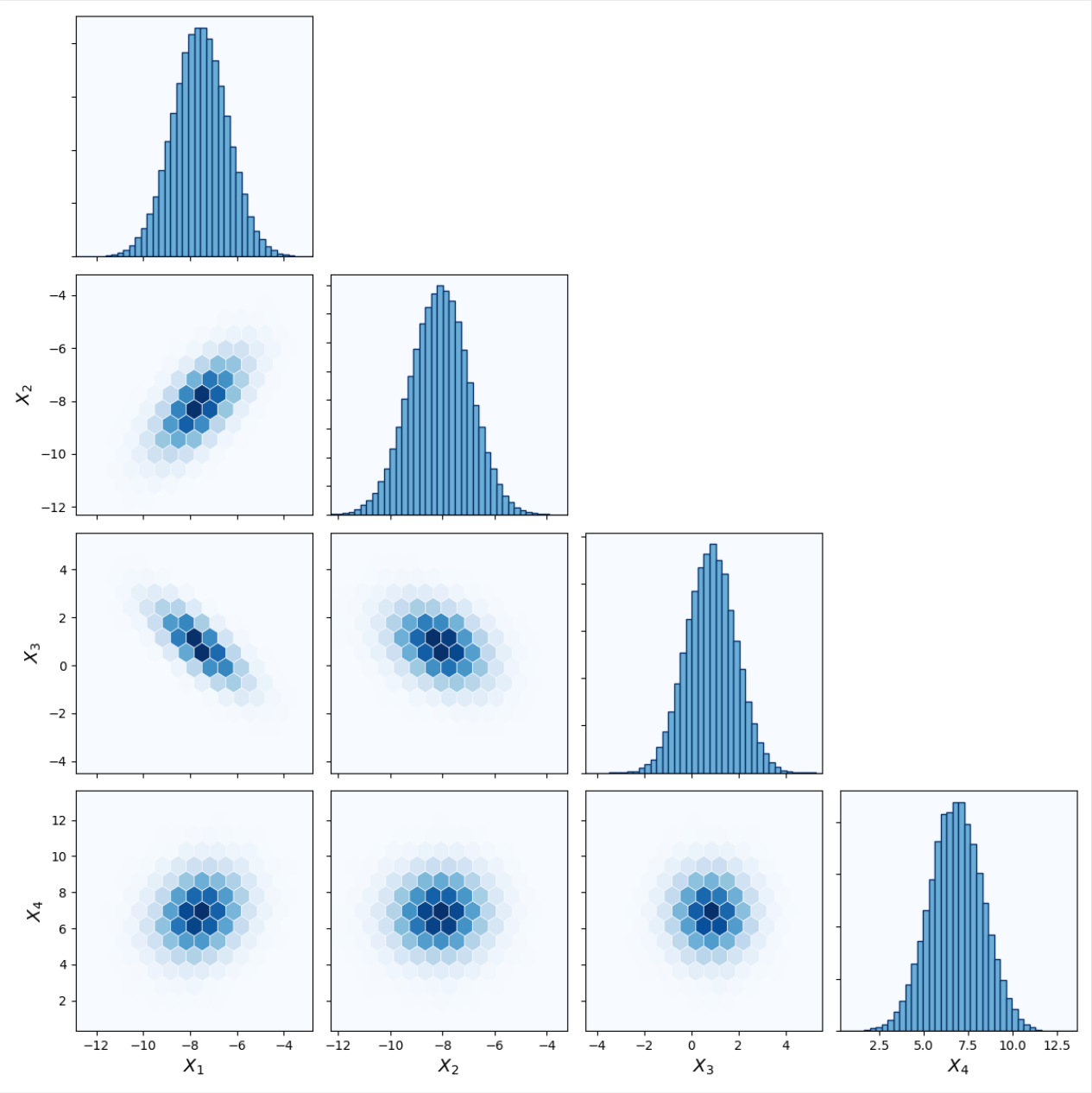
[5]:
```
cornerplot(
    data,
    labels=labels
);
```



1.2.2 Modifying number of bins

The number bins in the histgram and the number of hexbins in x-direction can be modified with the `hist_bins` and `hex_gridsize` keywords. Default values are 20 and 30 respectively.

[6]:
```
cornerplot(
    data,
    hex_gridsize=15,
    hist_bins=40,
    labels=labels
);
```



1.2.3 Changing axis limits

By default the axis limits are the minimum and maximum values of each feature respectively. Custom axis limits can be set with the `limits` keyword argument.

It can be a list with the minimum and maximum values of the axis limits that is applied to all features.

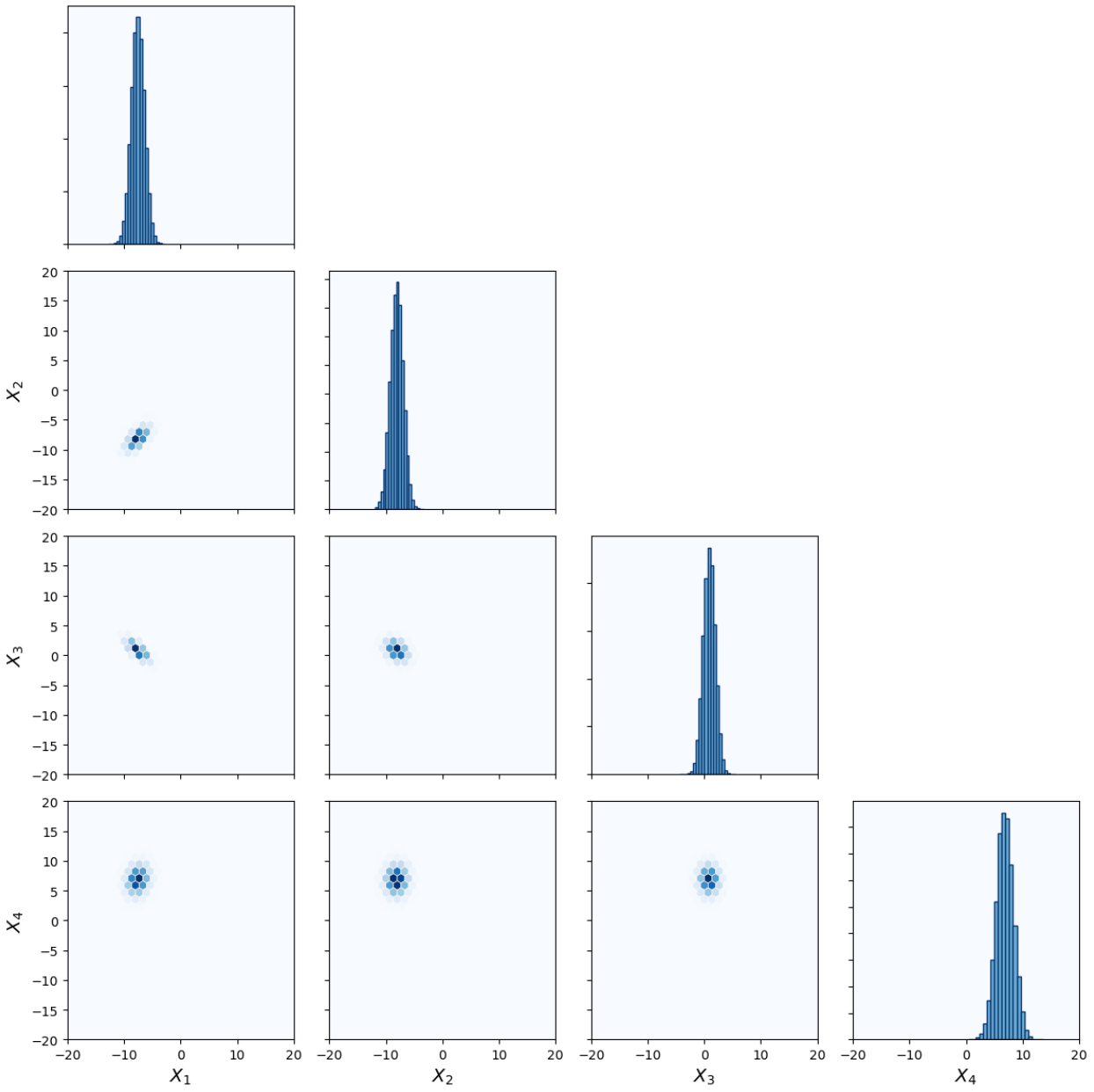
[7]:
```
limits = [-20., 20.]
```

[8]:
```
cornerplot(
    data,
```

(continues on next page)

(continued from previous page)

```
    labels=labels,
    limits=limits
);
```



It is furthermore possible to specify the limits for every feature separately.

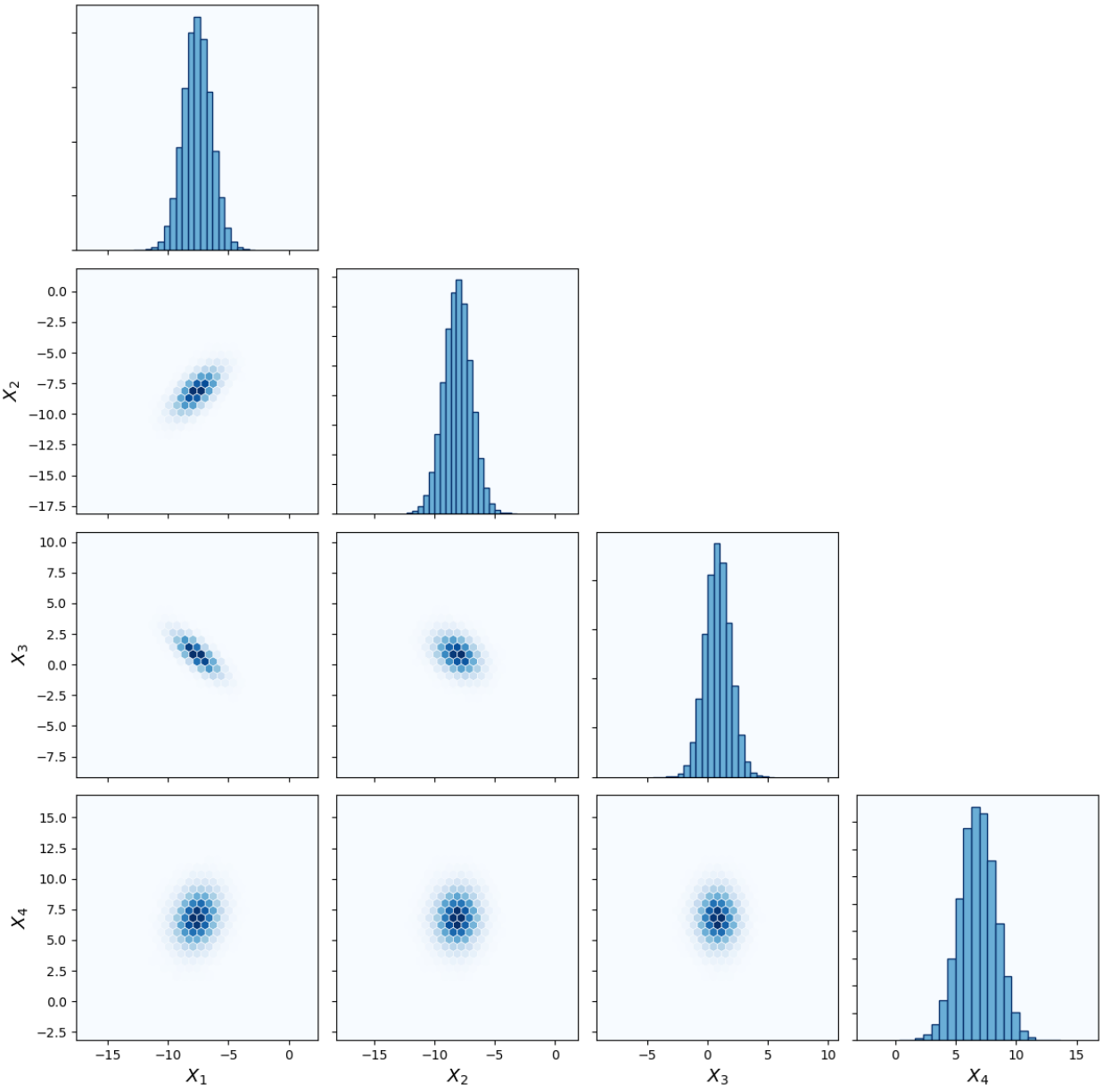
[9]:
```
limits = np.vstack((mean-10, mean+10)).T
```

[10]:
```
cornerplot(
    data,
    labels=labels,
    limits=limits
```

(continues on next page)

(continued from previous page)

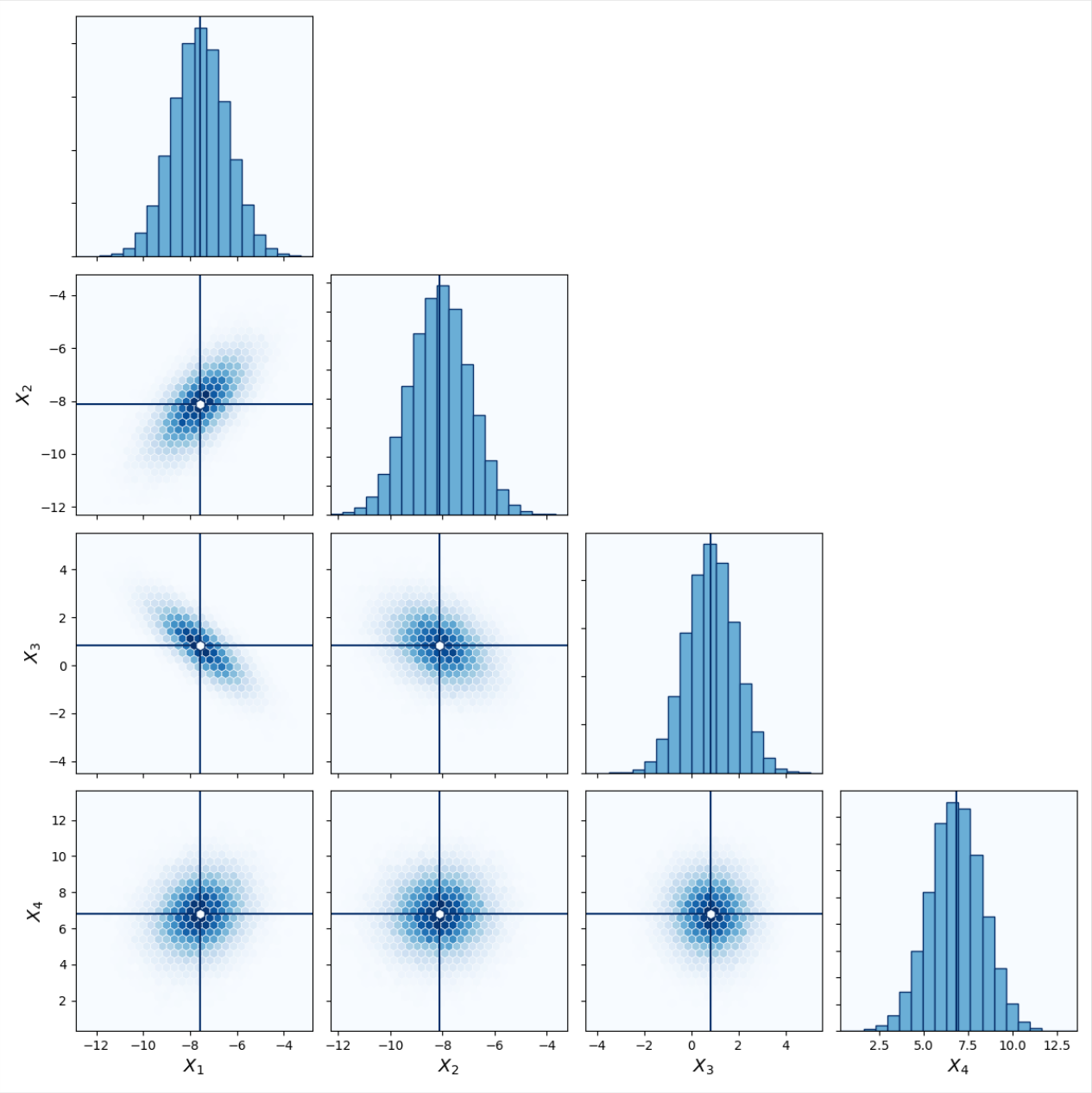
```
);
```



1.2.4 Highlighting specific values

It is often usefull to highlight specific values in the corner plot. In this example we want to highlight the true mean values of the features that we have randomly picked above.

[11]:
```
cornerplot(
    data,
    highlight=mean,
    labels=labels
);
```



1.2.5 Adding σ contour levels

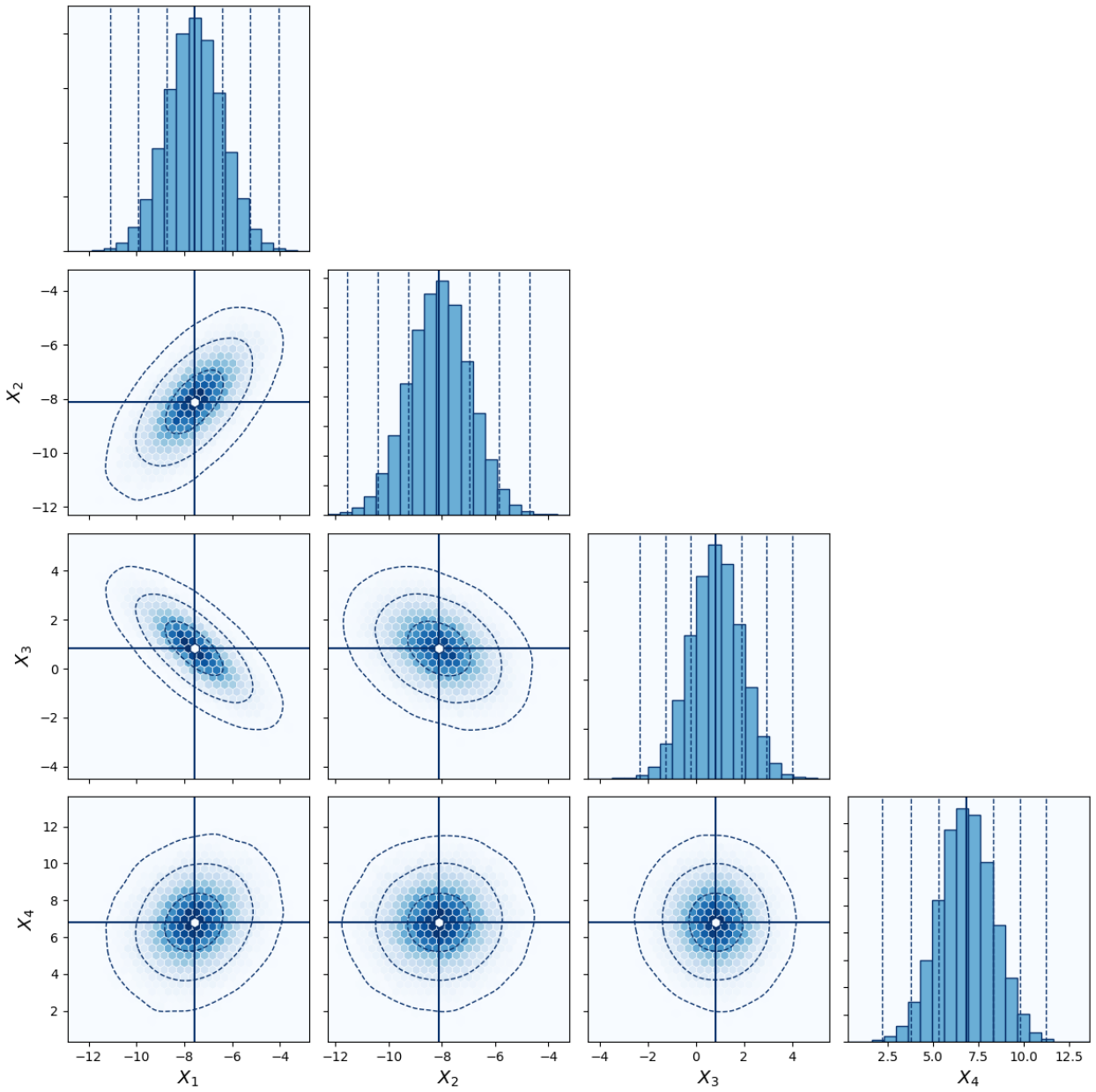
It is additionally possible to add contour lines of the σ levels of the data. In this example we want to display the 1σ and 2σ levels. Plotting the contour lines requires smoothing the hexbin histogram data. This can be accessed via the `sigma_smooth` keyword. default value is 3.

[12]:
```
cornerplot(
    data,
    highlight=mean,
    labels=labels,
    sigma_levels=[1, 2, 3],
```

(continues on next page)

(continued from previous page)

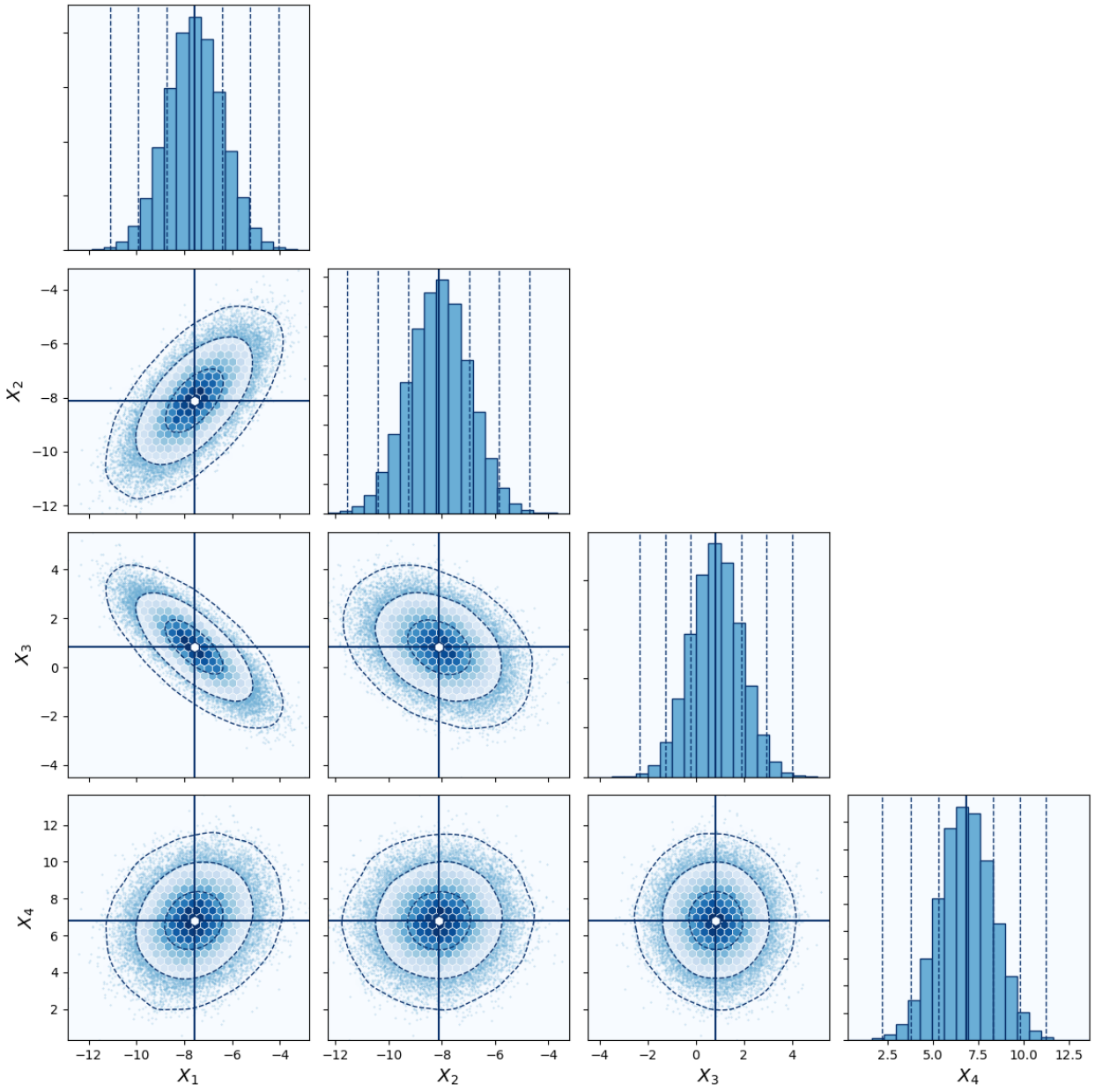
```
    sigma_smooth=3
);
```



1.2.6 Displaying scatter plot outside of σ contours

To show scatter plots of individual data points in low density regions, use the `scatter_outside_sigma` keyword argument, which only plots data points outside a given `\sigma` threshold.

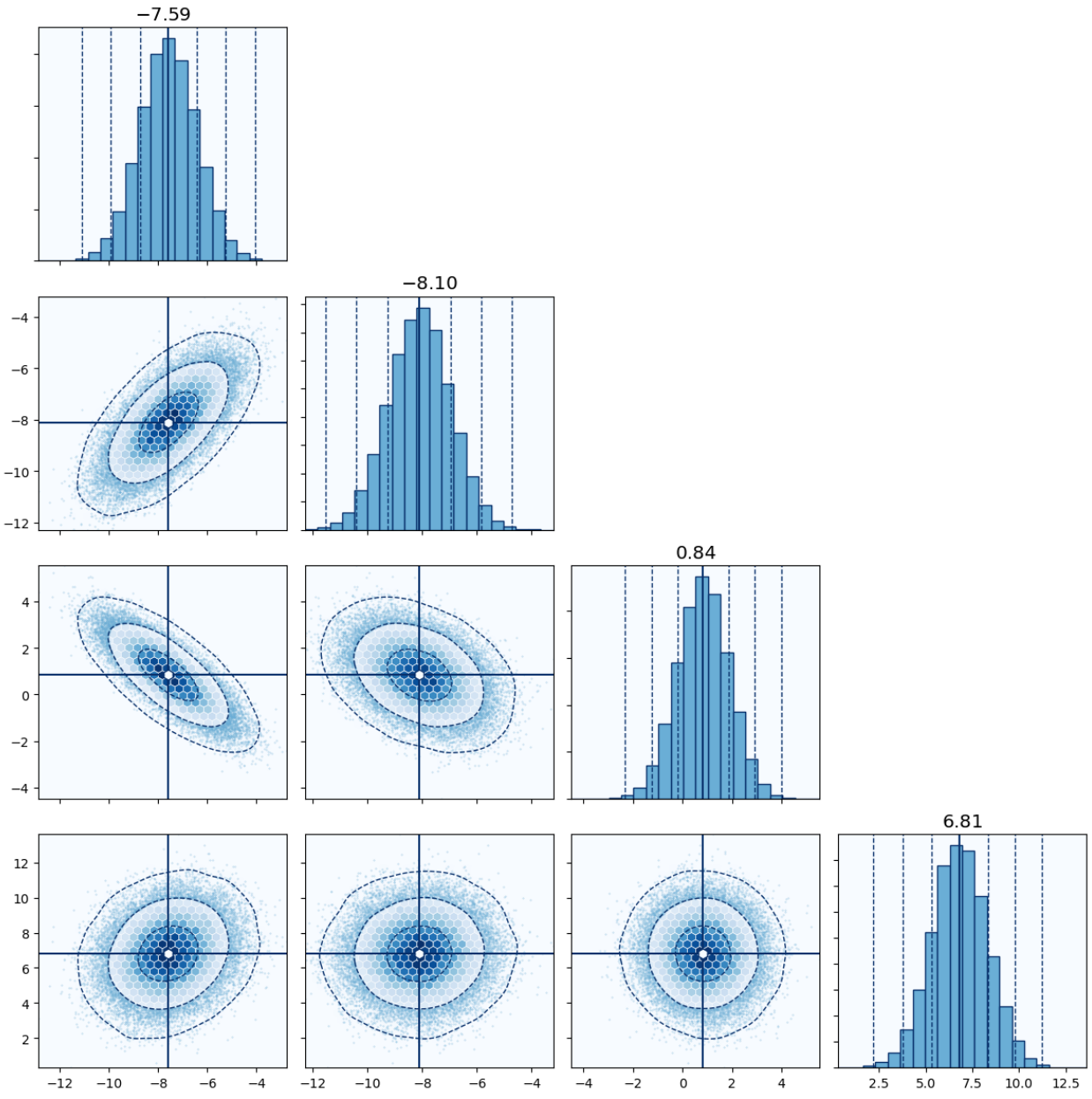
[13]:
```
cornerplot(
    data,
    highlight=mean,
    labels=labels,
    scatter_outside_sigma=2,
    sigma_levels=[1, 2, 3],
    sigma_smooth=3
);
```



1.2.7 Displaying quantiles

It is possible to specific quantiles as title of the histograms. In this example we want to display the mean values of the features computed from the dataset. The mean values is the `[50]` quantile.

[14]:
```
cornerplot(
    data,
    highlight=mean,
    scatter_outside_sigma=2,
    sigma_levels=[1, 2, 3],
    sigma_smooth=3,
    title_quantiles=[50]
);
```

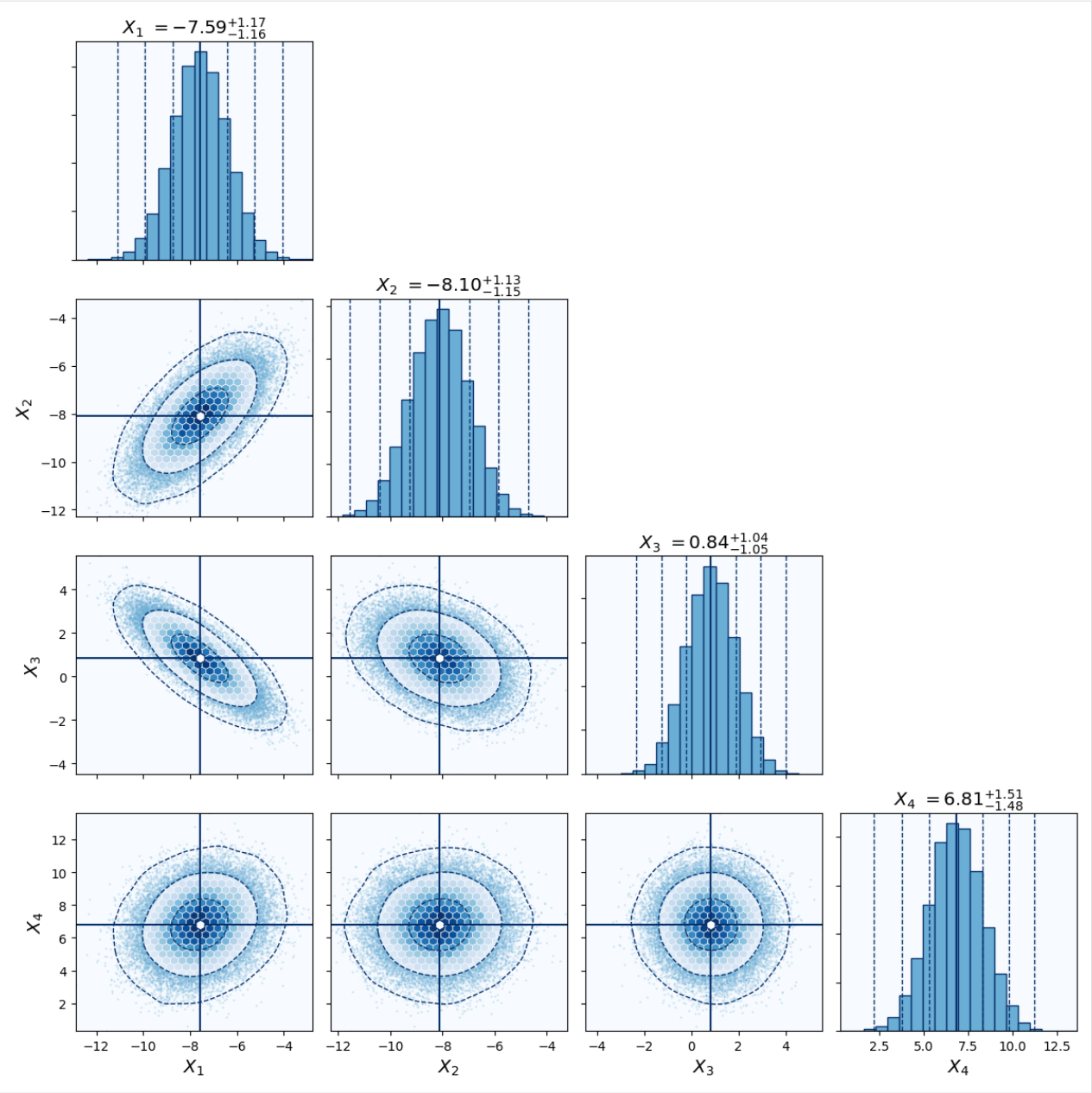


Additionally we can plot the 1σ intervals of the mean. For this we need to now what the 1σ quantiles of a normal distributions are. We utilize the helper function `cornerhex.sigma_to_quantile` for this purpose.

[15]:
```
from cornerhex import sigma_to_quantile

quantiles = [50-sigma_to_quantile(1.), 50, 50+sigma_to_quantile(1.)]
```

If the feature `labels` are given they will be displayed in the title as well.

[16]:
```
cornerplot(
    data,
    highlight=mean,
    labels=labels,
    scatter_outside_sigma=2,
    sigma_levels=[1, 2, 3],
    sigma_smooth=3,
    title_quantiles=quantiles
);
```



1.2.8 Displaying Pearson correlation coefficient

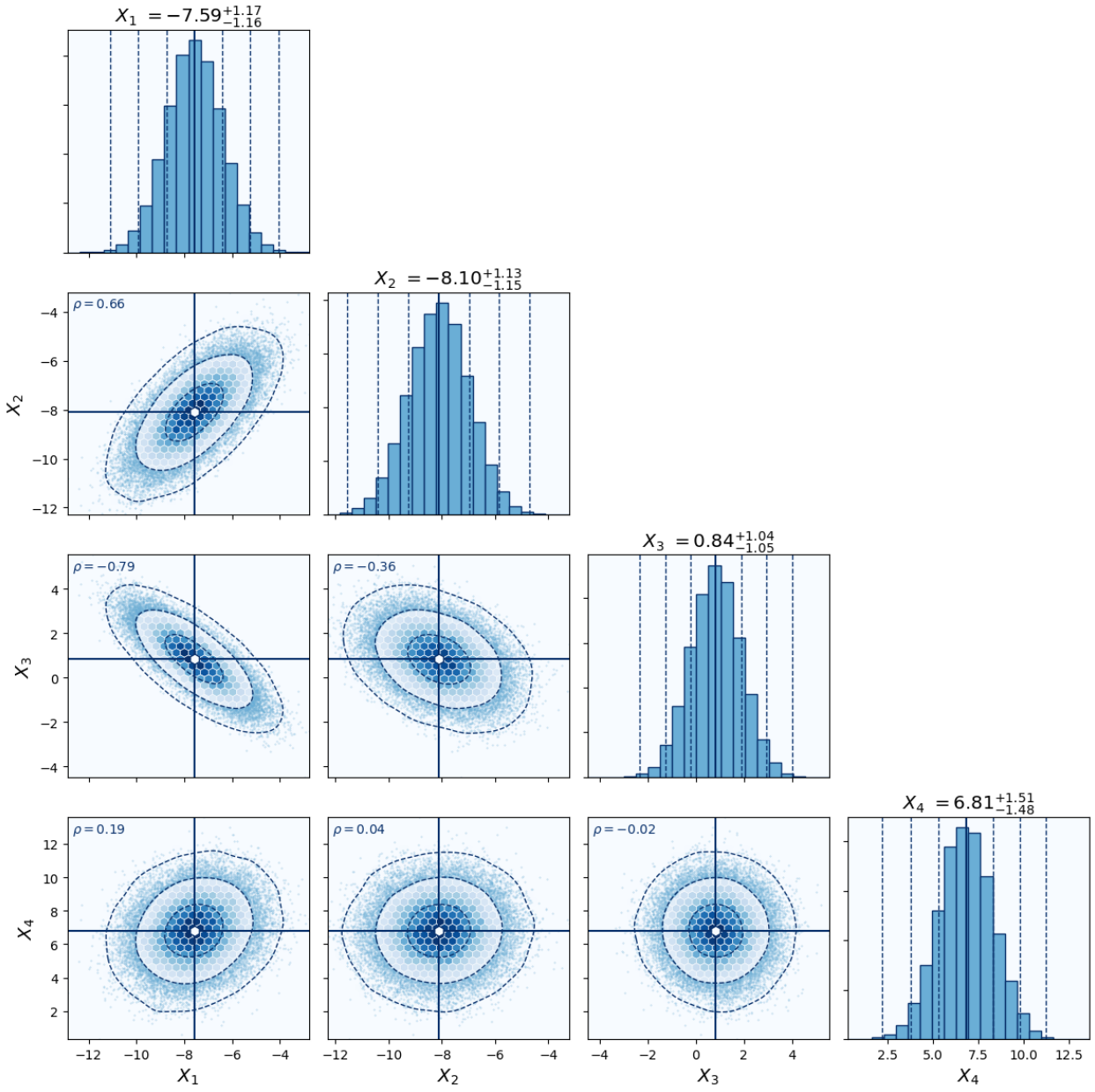
It is furthermore possible to display the Pearson correlation coefficient ρ in each hexbin plot with the `show_correlations` keyword.

[17]:
```
cornerplot(
    data,
    highlight=mean,
    labels=labels,
    show_correlations=True,
    scatter_outside_sigma=2,
```

(continues on next page)

(continued from previous page)

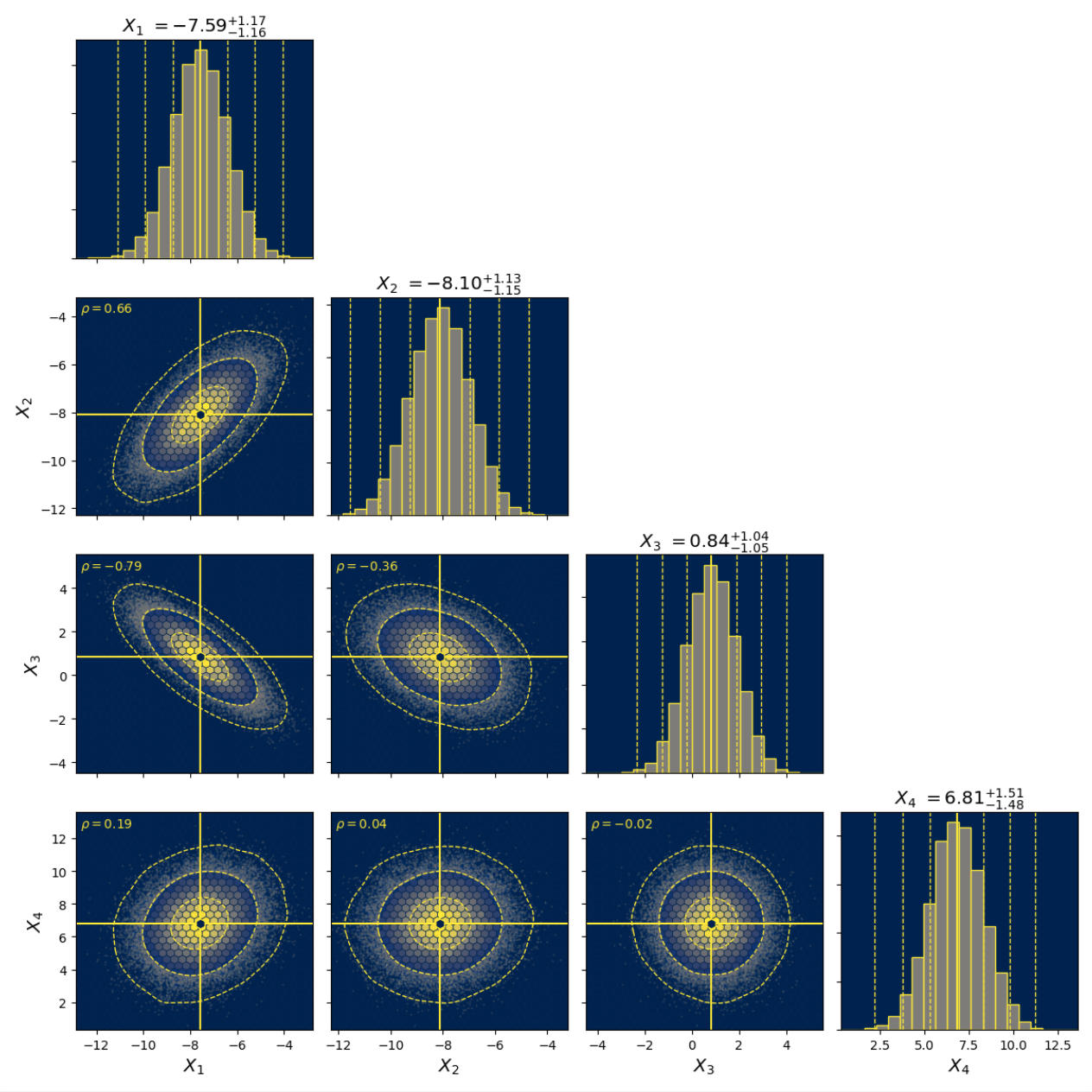
```
    sigma_levels=[1, 2, 3],
    sigma_smooth=3,
    title_quantiles=quantiles
);
```



1.2.9 Changing colormap

The colormap can be modified by either specifying a string with a valid name of a colormap or by passing a colormap object itself. The colors of line will be changed accordingly.

[18]:
```
cornerplot(
    data,
    cmap="cividis",
    highlight=mean,
    labels=labels,
    show_correlations=True,
    scatter_outside_sigma=2,
    sigma_levels=[1, 2, 3],
    sigma_smooth=3,
    title_quantiles=quantiles
);
```



1.2.10 Changing line and textcolor

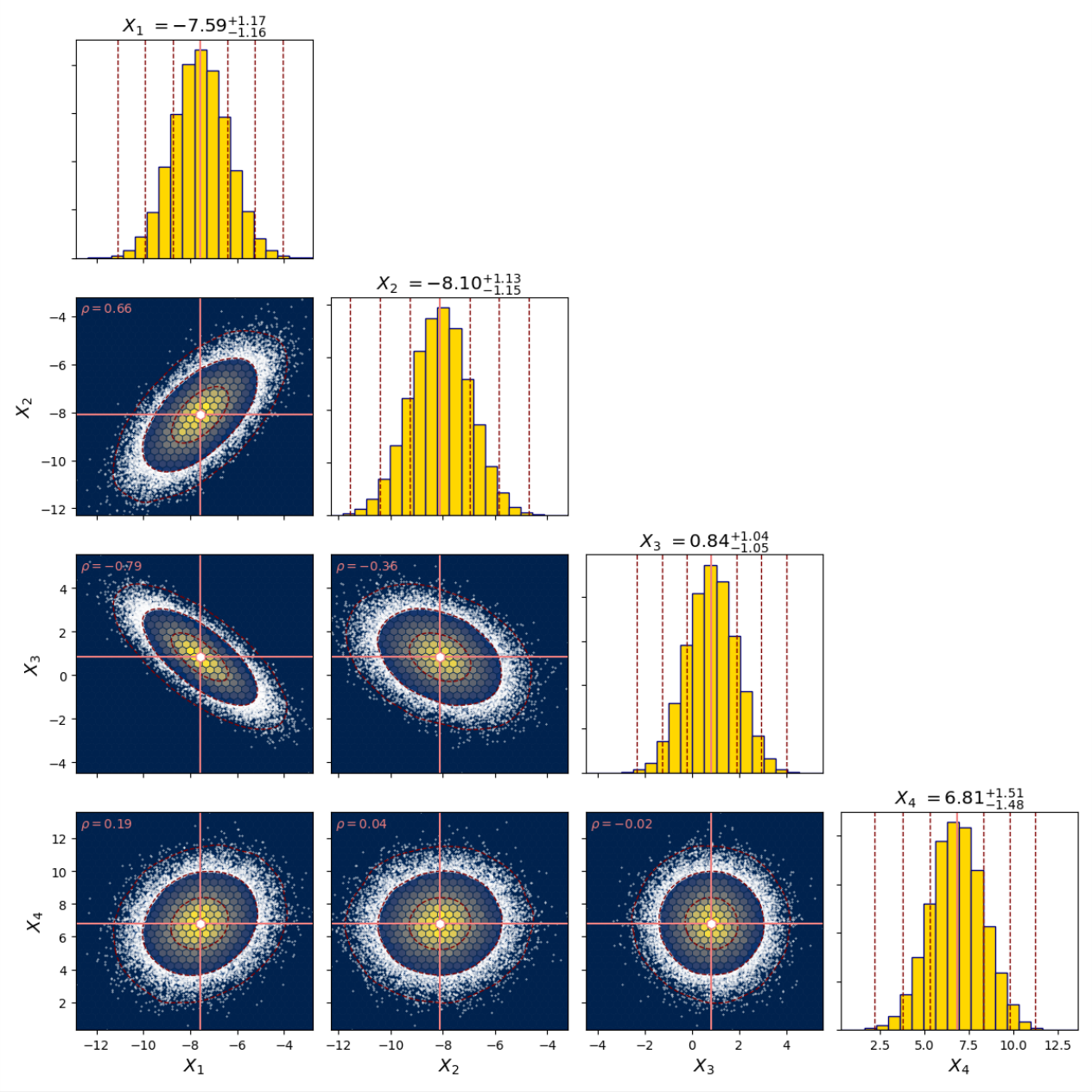
The edge-, face- and background colors of the histograms, the linecolors of the σ contours and the highlights, the color of the scatter points and their α-tranparency value, as well as the textcolor of the correlation coefficients adapt to the chosen colormap. Additionally they can be specified as well.

[19]:
```
cornerplot(
    data,
    cmap="cividis",
    correlation_textcolor="lightcoral",
    highlight=mean,
```

(continues on next page)

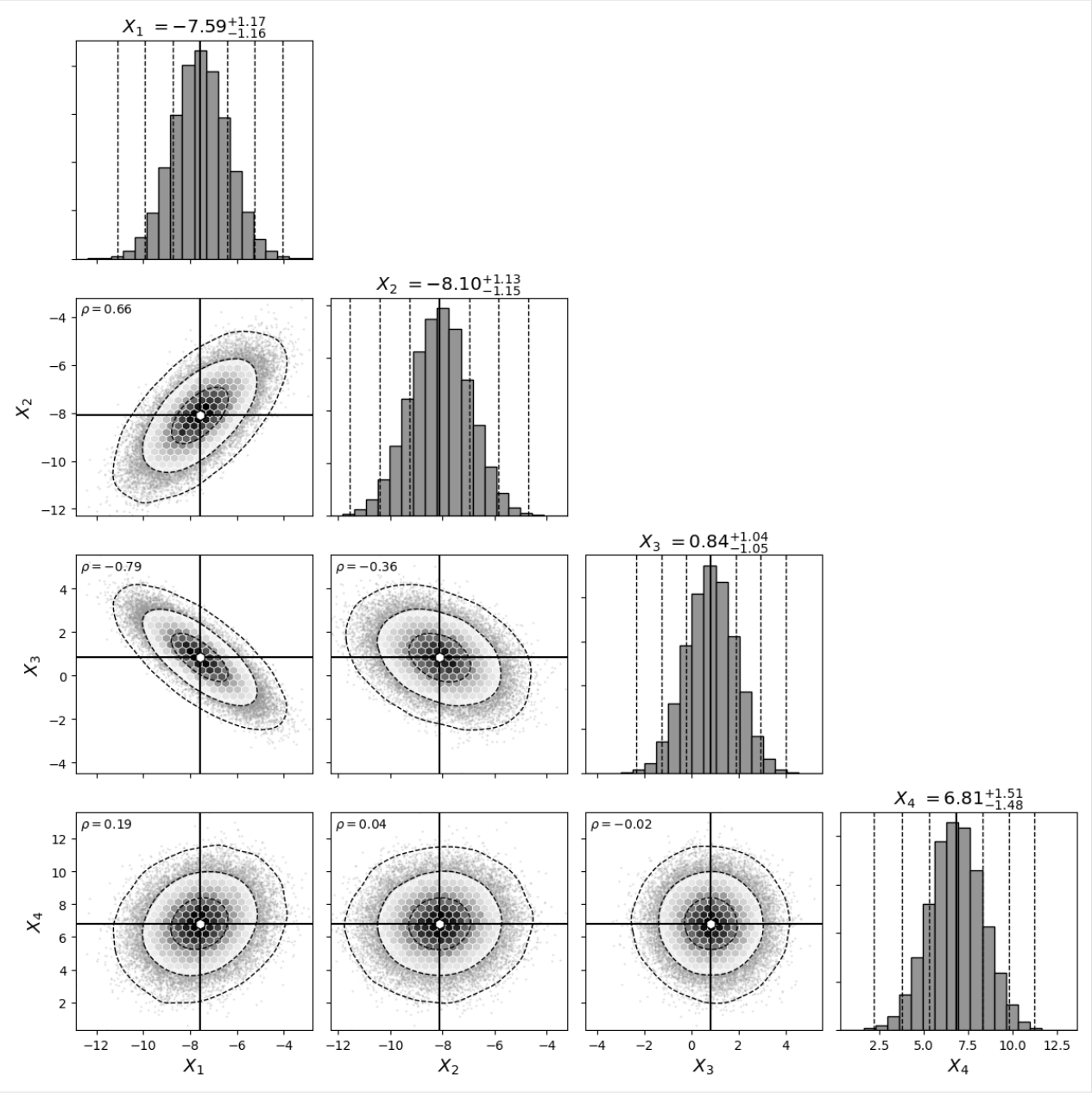
(continued from previous page)

```
    highlight_linecolor="lightcoral",
    highlight_markercolor="white",
    hist_backgroundcolor="white",
    hist_edgecolor="navy",
    hist_facecolor="gold",
    labels=labels,
    show_correlations=True,
    scatter_alpha=1.,
    scatter_markercolor="white",
    scatter_outside_sigma=2,
    sigma_levels=[1, 2, 3],
    sigma_linecolor="maroon",
    sigma_smooth=3,
    title_quantiles=quantiles
);
```



Just because you can change the colors, does not mean that you should. It is often best to use a simple color scheme.

[20]:
```
cornerplot(
    data,
    cmap="Greys",
    highlight=mean,
    labels=labels,
    show_correlations=True,
    scatter_outside_sigma=2,
    sigma_levels=[1, 2, 3],
    sigma_smooth=3,
    title_quantiles=quantiles
);
```



1.2.11 Saving plots

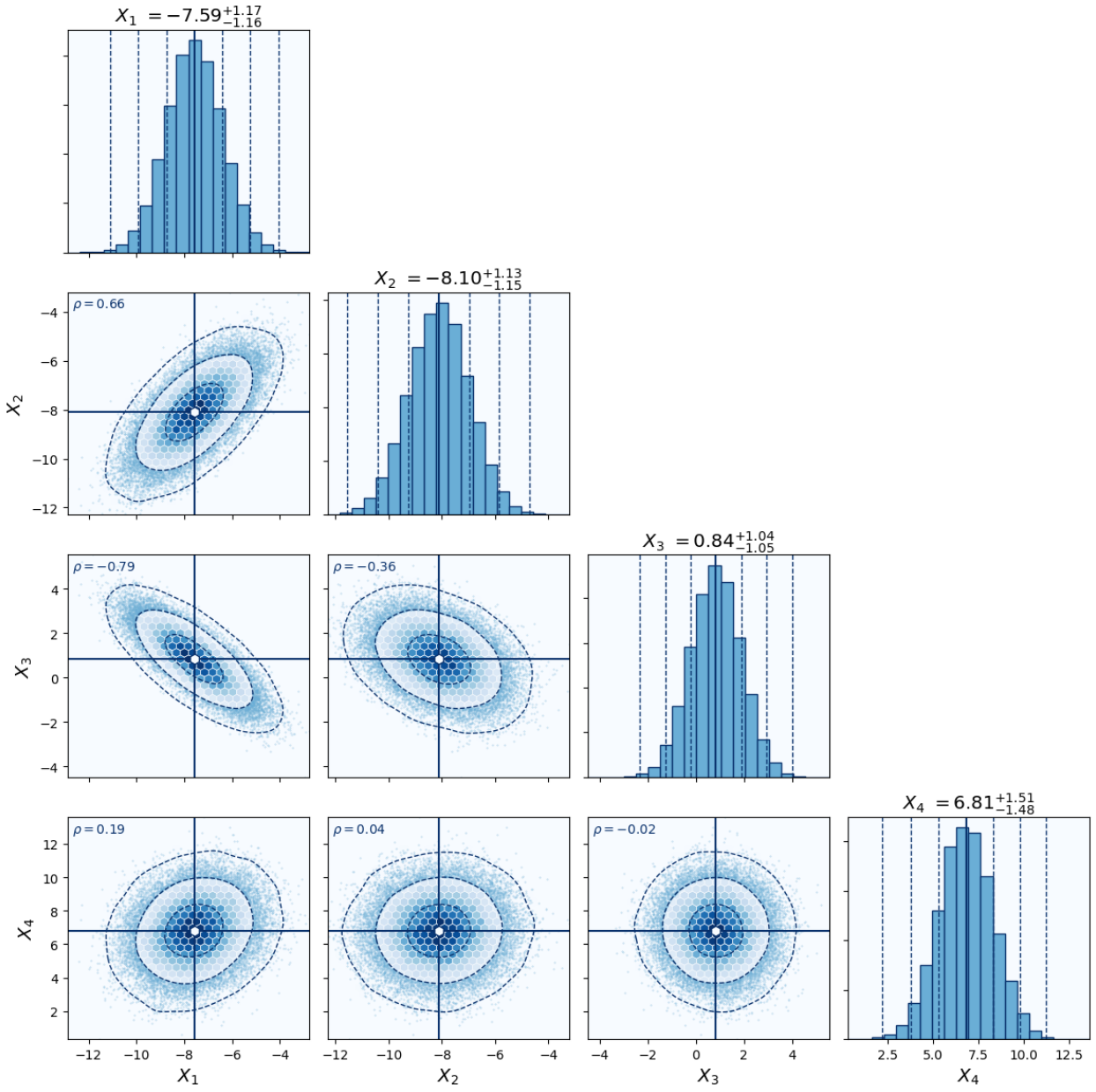
`cornerhex.cornerplot` returns the figure and axes objects of the plots, which can be used to further modify or to save the plot.

[21]:
```
fig, ax = cornerplot(
    data,
    highlight=mean,
    labels=labels,
    show_correlations=True,
    scatter_outside_sigma=2,
```

(continues on next page)

(continued from previous page)

```
    sigma_levels=[1, 2, 3],
    sigma_smooth=3,
    title_quantiles=quantiles
);
```



[22]:
```
fig.savefig("example.jpg")
```

1.3 Appendix A: Contributing/Bugs/Features

Before contributing to the software of `cornerhex`, please have a look at the Contribution Guidlines and the Code of Conduct.

1.3.1 Contributing

To contribute code please open a new pull request and describe the changes to the software your pull request introduces.

Please note, that additional features must also be described in the documentation.

1.3.2 Bug Reports

If you encounter a bug in `cornerhex`, please open a new bug report issue and describe the bug, the expected behavior, and steps how to reproduce the bug.

1.3.3 Feature Requests

If you have an idea of a new feature, that is missing in `cornerhex`, or if you want to suggest an improvement, please open a new feature request issue.

1.4 Module Reference

1.4.1 cornerhex Package

Functions

cornerplot(data[, cmap, ...])	Function creates hexbin corner plot matrix to visualize multidimensional data.
sigma_to_quantile(sig)	Function converts standard deviation of a normal distribution to quantile.

cornerplot

cornerhex.**cornerplot**(*data: ndarray*, *cmap='Blues'*, *correlation_textcolor=None*, *dpi=100.0*, *hex_gridsize=30*, *highlight=None*, *highlight_linecolor=None*, *highlight_markercolor=None*, *hist_backgroundcolor=None*, *hist_bins=20*, *hist_edgecolor=None*, *hist_facecolor=None*, *labels=None*, *limits=None*, *scatter_alpha=0.5*, *scatter_markercolor=None*, *scatter_outside_sigma=None*, *show_correlations=False*, *sigma_levels=None*, *sigma_linecolor=None*, *sigma_smooth=3.0*, *title_quantiles=None*, *width=3.0*) → Tuple[Figure, Axes]

Function creates hexbin corner plot matrix to visualize multidimensional data.

sigma_to_quantile

cornerhex.**sigma_to_quantile**(*sig: float*) → float

Function converts standard deviation of a normal distribution to quantile.

cornerhex is free to use. Please acknowledge the cornerhex repository if using it for publications.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

C

cornerhex, 23

INDEX

C

cornerhex
    module, 23
cornerplot() (*in module cornerhex*), 23

M

module
    cornerhex, 23

S

sigma_to_quantile() (*in module cornerhex*), 24