Branch: master ▾ oost_paper_code / error_ellipse.py Find file Copy path

joferkington Initial commit f3ca021 on Jun 8, 2012

1 contributor

82 lines (66 sloc) 2.62 KB Raw Blame History

```
1  import numpy as np
2
3  import matplotlib.pyplot as plt
4  from matplotlib.patches import Ellipse
5
6  def plot_point_cov(points, nstd=2, ax=None, **kwargs):
7      """
8      Plots an `nstd` sigma ellipse based on the mean and covariance of a point
9      "cloud" (points, an Nx2 array).
10
11     Parameters
12     ----------
13         points : An Nx2 array of the data points.
14         nstd : The radius of the ellipse in numbers of standard deviations.
15             Defaults to 2 standard deviations.
16         ax : The axis that the ellipse will be plotted on. Defaults to the
17             current axis.
```

```
18          Additional keyword arguments are pass on to the ellipse patch.
19
20      Returns
21      -------
22          A matplotlib ellipse artist
23      """
24      pos = points.mean(axis=0)
25      cov = np.cov(points, rowvar=False)
26      return plot_cov_ellipse(cov, pos, nstd, ax, **kwargs)
27
28  def plot_cov_ellipse(cov, pos, nstd=2, ax=None, **kwargs):
29      """
30      Plots an `nstd` sigma error ellipse based on the specified covariance
31      matrix (`cov`). Additional keyword arguments are passed on to the
32      ellipse patch artist.
33
34      Parameters
35      ----------
36          cov : The 2x2 covariance matrix to base the ellipse on
37          pos : The location of the center of the ellipse. Expects a 2-element
38              sequence of [x0, y0].
39          nstd : The radius of the ellipse in numbers of standard deviations.
40              Defaults to 2 standard deviations.
41          ax : The axis that the ellipse will be plotted on. Defaults to the
42              current axis.
43          Additional keyword arguments are pass on to the ellipse patch.
44
45      Returns
46      -------
47          A matplotlib ellipse artist
48      """
49      def eigsorted(cov):
50          vals, vecs = np.linalg.eigh(cov)
51          order = vals.argsort()[::-1]
52          return vals[order], vecs[:,order]
53
```

```
54      if ax is None:
55          ax = plt.gca()
56
57      vals, vecs = eigsorted(cov)
58      theta = np.degrees(np.arctan2(*vecs[:,0][::-1]))
59
60      # Width and height are "full" widths, not radius
61      width, height = 2 * nstd * np.sqrt(vals)
62      ellip = Ellipse(xy=pos, width=width, height=height, angle=theta, **kwargs)
63
64      ax.add_artist(ellip)
65      return ellip
66
67  if __name__ == '__main__':
68      #-- Example usage -----------------------
69      # Generate some random, correlated data
70      points = np.random.multivariate_normal(
71              mean=(1,1), cov=[[0.4, 9],[9, 10]], size=1000
72              )
73      # Plot the raw points...
74      x, y = points.T
75      plt.plot(x, y, 'ro')
76
77      # Plot a transparent 3 standard deviation covariance ellipse
78      plot_point_cov(points, nstd=3, alpha=0.5, color='green')
79
80      plt.show()
81
```