

Listing 1 "Kernel Fourier transforms"

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 import math
4
5 from matplotlib import rc
6 rc('font',**{'family':'sans-serif','sans-serif':['Helvetica']})
7 ## for Palatino and other serif fonts use:
8 #rc('font',**{'family':'serif','serif':['Palatino']})
9 rc('text', usetex=True)
10
11 # del matplotlib.font_manager.weight_dict['roman']
12 # matplotlib.font_manager._rebuild()
13
14 def Epanechnikov_Kernel_FT(u):
15     result = []
16     for _u in u:
17         if abs(_u) < 0.01:
18             v = 1
19         else:
20             v = 3 * ( np.sin(_u) - _u * np.cos(_u) ) / pow(_u, 3)
21         result.append(v)
22     return result
23
24 def Spline_Kernel_FT(u, beta):
25     result = []
26     for _u in u:
27         v = 1 / ( 1 + pow(abs(_u), beta))
28         result.append(v)
29     return result
30
31 def Pinsker_Kernel_FT(u, beta):
32     result = []
33     for _u in u:
34         v = max(0, ( 1 - pow(abs(_u), beta)))
35         result.append(v)
36     return result
37
38 def Silverman_Kernel_FT(u):
39     result = []
40     for _u in u:
41         v = 1 / ( 1 + pow(_u, 4))
42         result.append(v)
43     return result
44
45 def Pinsker_Kernel(u):
46     result = []
47     for _u in u:
48         if abs(_u) < 0.01:
49             v = 2 / (3 * np.pi)
50         else:
51             v = 2 * ( np.sin(_u) - _u * np.cos(_u) ) / ( np.pi * pow(_u, 3) )
52         result.append(v)
53     return result
54
55
56 def main():
57     plt.style.use('_mpl-gallery')
58
59     x_min = -20
60     x_max = 20
61     x_pitch = 0.01
62
63     u = np.arange(x_min, x_max, x_pitch)
64
65     ft_Epane = Epanechnikov_Kernel_FT(u)
66     ft_Silverman = Silverman_Kernel_FT(u)
67
68     beta = 2
69     ft_Spline = Spline_Kernel_FT(u, beta)

```

```

70     ft_Pinsker = Pinsker_Kernel_FT(u, beta)
71
72     fig, ax = plt.subplots()
73
74     ax.set_title('Kernel_FT', fontsize=18)
75     ax.set_xlabel(r'$\omega$', fontsize=14)
76     ax.set_ylabel(r'$\hat{K}(\omega)$', fontsize=14)
77
78     ax.plot(u, ft_Epane, color='b', linewidth=3.0, label="Epanechnikov_Kernel_[p25]")
79     ax.plot(u, ft_Silverman, color='g', linewidth=3.0, label="Silverman_Kernel_[p27]")
80     ax.plot(u, ft_Spline, color='r', linewidth=3.0, label="Spline_Kernel_(1.55)_[p27]")
81     ax.plot(u, ft_Pinsker, color='m', linewidth=3.0, label="Pinsker_Kernel_(1.56)_[p27]")
82
83     x_tick = 1
84     y_tick = 0.1
85
86     ft = ft_Epane + ft_Silverman + ft_Spline + ft_Pinsker
87
88     y_min = math.floor(min(ft)*10) / 10 - y_tick
89     y_max = math.ceil(max(ft)*10) / 10 + y_tick
90
91     ax.set(xlim=(x_min, x_max), xticks=np.arange(x_min, x_max, x_tick), ylim=(y_min, y_max), yticks=np.
92           arange(y_min, y_max, y_tick))
93     plt.subplots_adjust(left=0.1, bottom=0.1, top=0.9)
94
95     ax.axhline(0, color='k', linewidth=1.0)
96     ax.axvline(0, color='k', linewidth=1.0)
97
98     plt.legend(fontsize=14)
99
100     ep_tex = r'\begin{eqnarray*}\hat{K}(\omega)_{\omega}=\frac{3}{\omega^3}(\sin\omega-\omega\cos\omega)\end{eqnarray*}'
101     ax.text(x_min + (x_max - x_min) / 2 + 7 * x_tick, y_min + (y_max - y_min) / 2 + y_tick, ep_tex, color=
102            "b", fontsize=20)
103
104     slv_tex = r'\begin{eqnarray*}\hat{K}(\omega)_{\omega}=\frac{1}{1+\omega^4}\end{eqnarray*}'
105     ax.text(x_min + (x_max - x_min) / 2 + 7 * x_tick, y_min + (y_max - y_min) / 2, slv_tex, color="g",
106            fontsize=20)
107
108     spl_tex = r'\begin{eqnarray*}\hat{K}(\omega)_{\omega}=\frac{1}{1+|\omega|^{\beta(=2)}}\end{eqnarray*}'
109     ax.text(x_min + (x_max - x_min) / 2 + 7 * x_tick, y_min + (y_max - y_min) / 2 - y_tick, spl_tex, color=
110            "r", fontsize=20)
111
112     pin_tex = r'\begin{eqnarray*}\hat{K}(\omega)_{\omega}=(1-|\omega|^{\beta(=2)})_{+}\end{eqnarray*}'
113     ax.text(x_min + (x_max - x_min) / 2 + 7 * x_tick, y_min + (y_max - y_min) / 2 - 2 * y_tick, pin_tex,
114            color="m", fontsize=20)
115
116     plt.show()
117
118 if __name__ == "__main__":
119     main()

```