test_unbinned_fit

October 19, 2021

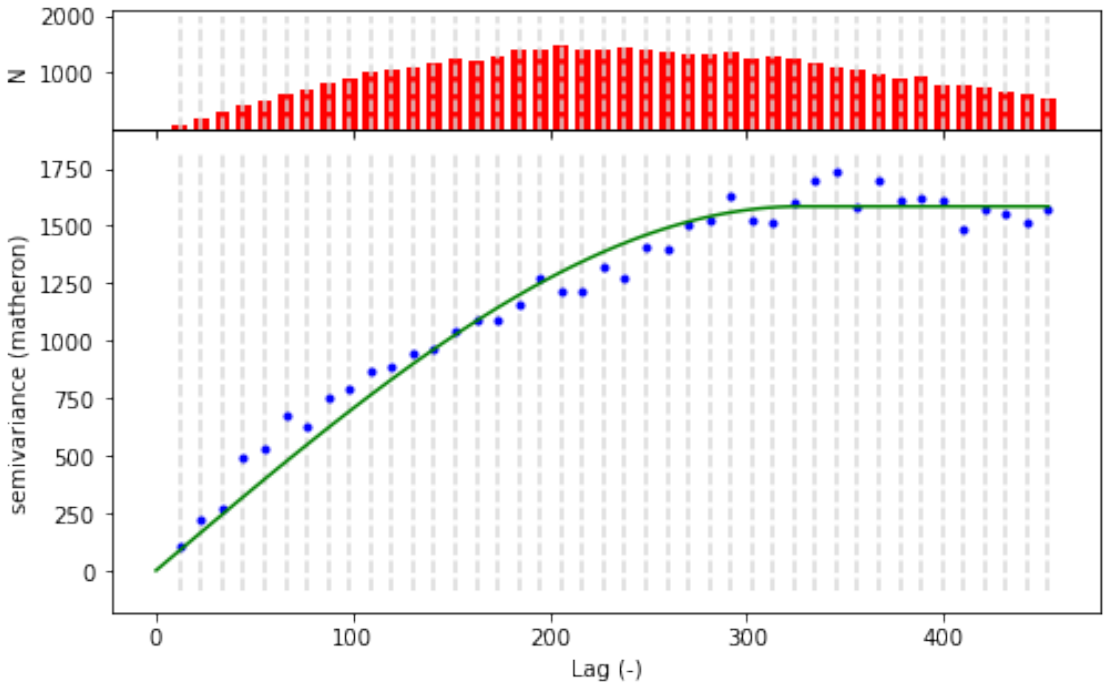
0.1 Test unbinned fit

[1]:
```
import skgstat as skg
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import inv, det
from scipy.spatial.distance import squareform
import warnings
from time import time
warnings.filterwarnings('ignore')
```

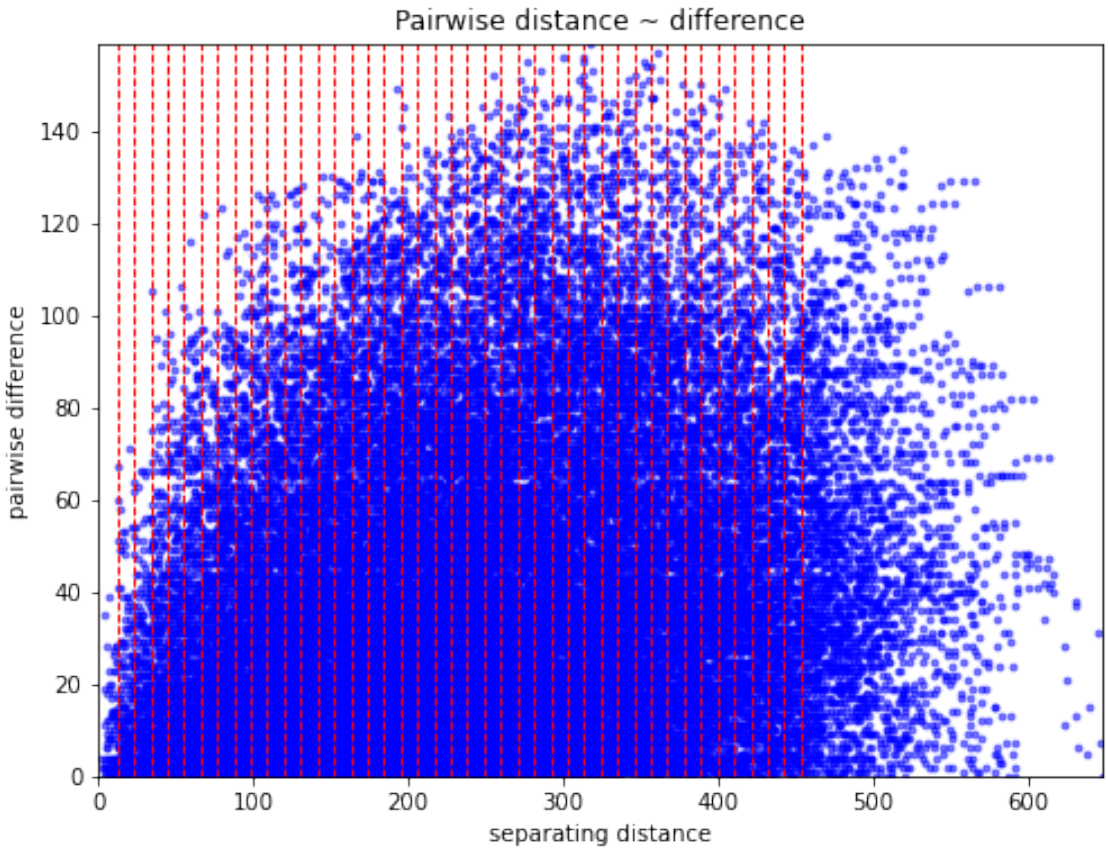
[2]:
```
# use the same dataset as used in GMD paper
c, v = skg.data.pancake(N=300, seed=42).get('sample')
```

[9]:
```
t1 = time()
V = skg.Variogram(c,v, bin_func='scott', maxlag=0.7)
t2 = time() # get time for full analysis, including fit
print(f"Processing time: {round((t2 - t1) * 1000)} ms")
print(V)
fig = V.plot()
```

```
Processing time: 14 ms
spherical Variogram
-------------------
Estimator:         matheron
Effective Range:   326.72
Sill:              1584.49
Nugget:            0.00
```



[4]:
```
fig = V.distance_difference_plot(show=False)
```



0.2 ML

Implementing eq. 14 from Lark (2000), with spherical model, adapted to fit eq. 16 from same publication.

[5]:
```
def f(h, a):
    if h >= a:
        return 1
    elif h == 0:
        return 0
    return (3*h) / (2*a) - 0.5 * (h / a)**3

def get_A(r, s, b, dists):
    a = np.array([f(d, r) for d in dists])
    A = squareform((s / (s + b)) * (1 - a))
    np.fill_diagonal(A, 1)

    return A

def like(r, s, b, z, dists):
```

```
    A = get_A(r, s, b, dists)
    n = len(A)
    A_inv = inv(A)
    ones = np.ones((n, 1))
    z = z.reshape(n, -1)
    m = inv(ones.T @ A_inv @ ones) @ (ones.T @ A_inv @ z)
    b = np.log((z - m).T @ A_inv @ (z - m))
#    print(b)
    d = np.log(det(A))
#    print(d)
    loglike = (n / 2)*np.log(2*np.pi) + (n / 2) - (n / 2)* np.log(n) + 0.5* d +␣
 ↪(n / 2) * b
    return loglike.flatten()[0]
```

[6]:
```
from scipy.optimize import minimize

z = V.values#.reshape(len(V.values), -1)
dists = V.distance
fun = lambda x, *args: like(x[0], x[1], x[2], z=z, dists=dists)
t3 = time()
res = minimize(fun, [np.mean(V.distance), np.var(V.values), 0.1 * np.var(V.
 ↪values)], bounds=[[0, V.bins[-1]], [0, 2500], [0, 2400]])
t4 = time()
print(f"Processing time {np.round(t4 - t3, 2)} seconds")
```

```
Processing time 2.14 seconds
```

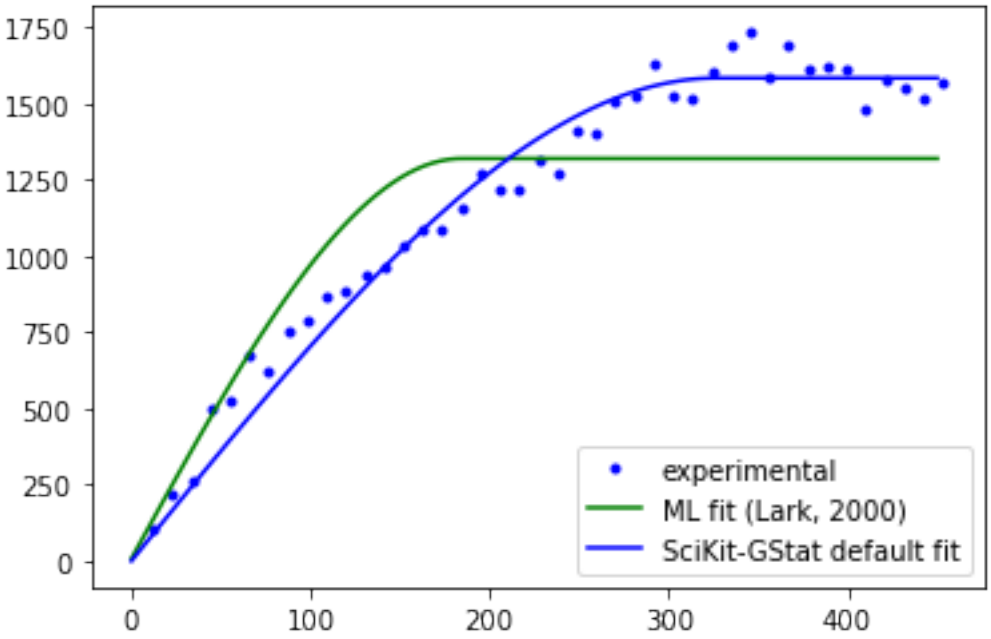
[7]:
```
print('p0: ', [np.mean(V.distance), np.var(V.values), 0.1 * np.var(V.values)])
print('res:', res.x)
```

```
p0:  [253.54190639540656, 1298.8712333333333, 129.88712333333334]
res: [ 184.23866253 1312.89640482    7.23090537]
```

[8]:
```
import matplotlib.pyplot as plt
mod = lambda h: f(h, res.x[0]) * res.x[1] + res.x[2]

x = np.linspace(0, 450, 100)
y = list(map(mod, x))
y2 = V.fitted_model(x)

plt.plot(V.bins, V.experimental, '.b', label='experimental')
plt.plot(x, y, '-g', label='ML fit (Lark, 2000)')
plt.plot(x, y2, '-b', label='SciKit-GStat default fit')
plt.legend(loc='lower right')
plt.gcf().savefig('compare.pdf', dpi=300)
```


1750
1500
1250
1000
750
500
250
0
0
100
200
300
400
experimental
ML fit (Lark, 2000)
SciKit-GStat default fit

[ ]: