semivarience

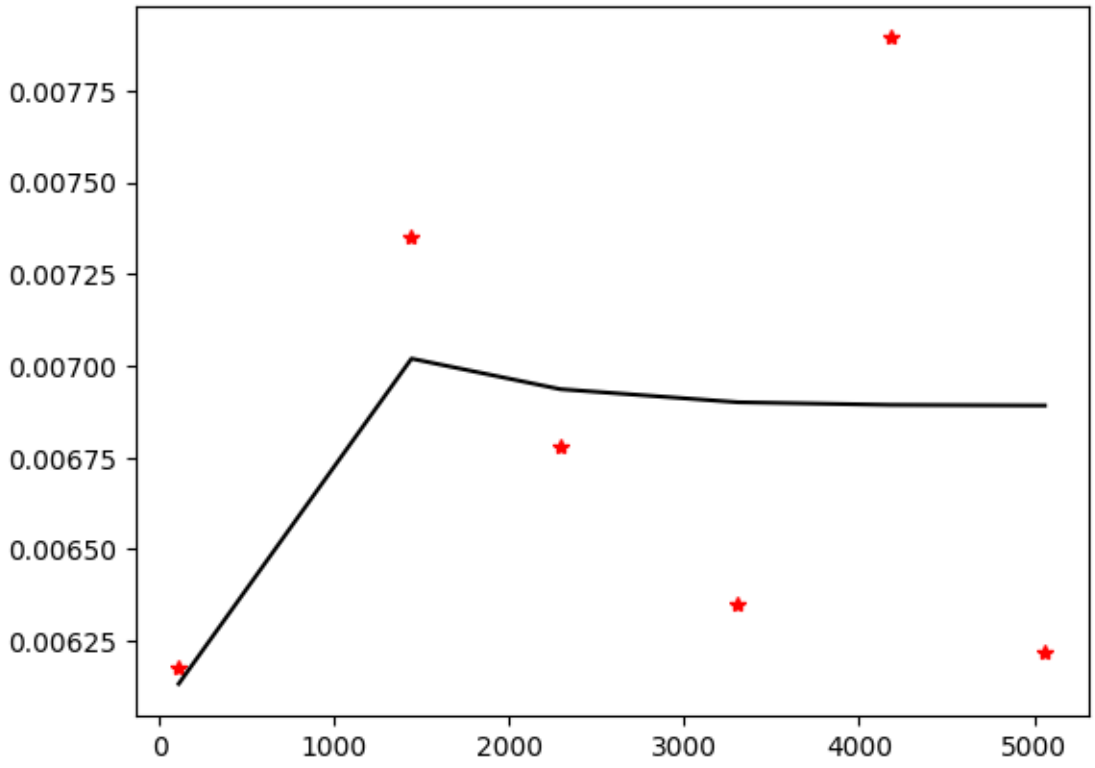
May 13, 2024

[1]:
```
import pandas as pd
from loadAndPreprocess import load_and_preprocess
from pykrige.ok3d import OrdinaryKriging3D
import pickle

'''
well_info: Well, X, Y, Total Resources
sensor_data: Depth, Porosity, Hydrate Saturation, Estimated Resources
'''
well_info, sensor_data_list = load_and_preprocess()

# Remove the data point with NaN value
well_info = well_info.dropna()
for sensor_data in sensor_data_list:
    sensor_data.dropna(inplace=True)


def compose_krige3D(name: str, sample_frac: float, variogram: str):
    # Make kriging interpolation for Porosity  individually
    data_frames = []
    for i, df in enumerate(sensor_data_list):
        df['X'] = well_info.loc[i, 'X']
        df['Y'] = well_info.loc[i, 'Y']
        df['Well'] = well_info.loc[i, 'Well']
        df['Total Resources'] = well_info.loc[i, 'Total Resources']
        data_frames.append(df)

    combined_data = pd.concat(data_frames, ignore_index=True)
    sampled_data = combined_data.sample(frac=sample_frac, random_state=1)  #␣
 ↪random_state for reproducibility

    X = sampled_data['X'].values
    Y = sampled_data['Y'].values
    Z = sampled_data['Depth'].values
    values = sampled_data[name].values

    # Create the 3D Kriging model
```

```
    ok3d = OrdinaryKriging3D(
        X,
        Y,
        Z,
        values,
        variogram_model=variogram,  # You might need to experiment with␣
↪different models
        verbose=True,
        enable_plotting=True,
    )

    return ok3d
```

[2]:
```
ok3d_poro = compose_krige3D("Porosity", 0.3, 'hole-effect')
```

```
Plotting Enabled

Adjusting data for anisotropy…
Initializing variogram model…
Using 'hole-effect' Variogram Model
Partial Sill: 0.0012084382568926166
Full Sill: 0.006891124882340032
Range: 1507.7406234905316
Nugget: 0.005682686625447415
```



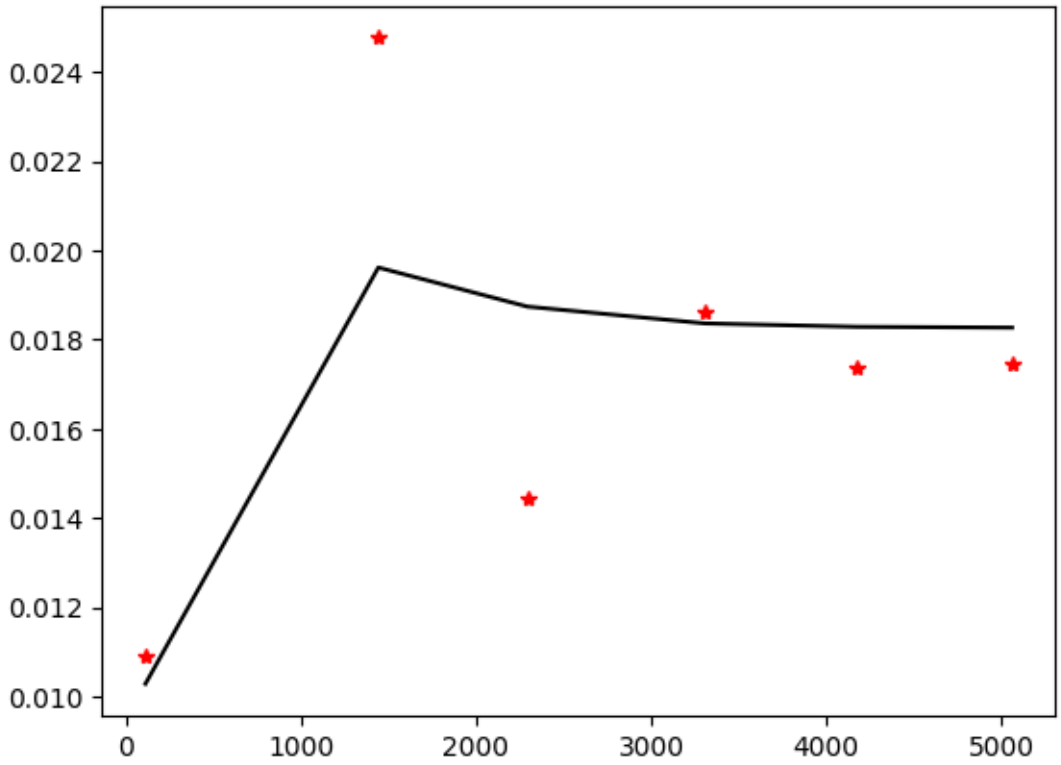
```
Calculating statistics on variogram model fit…
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[2], line 1
----> 1 ok3d_poro = compose_krige3D("Porosity", 0.3, 'hole-effect')

Cell In[1], line 37, in compose_krige3D(name, sample_frac, variogram)
     34 values = sampled_data[name].values
     36 # Create the 3D Kriging model
---> 37 ok3d = OrdinaryKriging3D(
     38     X,
     39     Y,
     40     Z,
     41     values,
     42␣
  ↪    variogram_model=variogram,  # You might need to experiment with different models
     43     verbose=True,
     44     enable_plotting=True,
     45 )
     47 return ok3d

File ~/model-2024c/.venv/lib/python3.12/site-packages/pykrige/ok3d.py:352, in␣
 ↪OrdinaryKriging3D.__init__(self, x, y, z, val, variogram_model,␣
 ↪variogram_parameters, variogram_function, nlags, weight, anisotropy_scaling_y␣
 ↪anisotropy_scaling_z, anisotropy_angle_x, anisotropy_angle_y,␣
 ↪anisotropy_angle_z, verbose, enable_plotting, exact_values, pseudo_inv,␣
 ↪pseudo_inv_type)
    350 if self.verbose:
    351     print("Calculating statistics on variogram model fit…")
--> 352 self.delta, self.sigma, self.epsilon = _find_statistics(
    353     np.vstack((self.X_ADJUSTED, self.Y_ADJUSTED, self.Z_ADJUSTED)).T,
    354     self.VALUES,
    355     self.variogram_function,
    356     self.variogram_model_parameters,
    357     "euclidean",
    358     self.pseudo_inv,
    359 )
    360 self.Q1 = core.calcQ1(self.epsilon)
    361 self.Q2 = core.calcQ2(self.epsilon)

File ~/model-2024c/.venv/lib/python3.12/site-packages/pykrige/core.py:807, in␣
 ↪_find_statistics(X, y, variogram_function, variogram_model_parameters,␣
 ↪coordinates_type, pseudo_inv)
    804     continue
    806 else:
```

```
--> 807     k, ss = _krige(
    808         X[:i, :],
    809         y[:i],
    810         X[i, :],
    811         variogram_function,
    812         variogram_model_parameters,
    813         coordinates_type,
    814         pseudo_inv,
    815     )
    817     # if the estimation error is zero, it's probably because
    818     # the evaluation point X[i, :] is really close to one of the
    819     # kriging system points in X[:i, :]…
    820     # in the case of zero estimation error, the results are not stored
    821     if np.absolute(ss) < eps:

File ~/model-2024c/.venv/lib/python3.12/site-packages/pykrige/core.py:750, in␣
 ↪_krige(X, y, coords, variogram_function, variogram_model_parameters,␣
 ↪coordinates_type, pseudo_inv)
    748     res = np.linalg.lstsq(a, b, rcond=None)[0]
    749 else:
--> 750     res = np.linalg.solve(a, b)
    751 zinterp = np.sum(res[:n, 0] * y)
    752 sigmasq = np.sum(res[:, 0] * -b[:, 0])

File ~/model-2024c/.venv/lib/python3.12/site-packages/numpy/linalg/linalg.py:
 ↪409, in solve(a, b)
    407 signature = 'DD->D' if isComplexType(t) else 'dd->d'
    408 extobj = get_linalg_error_extobj(_raise_linalgerror_singular)
--> 409 r = gufunc(a, b, signature=signature, extobj=extobj)
    411 return wrap(r.astype(result_t, copy=False))

KeyboardInterrupt:
```

[3]:
```
ok3d_hydr = compose_krige3D("Hydrate Saturation", 0.3, 'hole-effect')
```

```
Plotting Enabled

Adjusting data for anisotropy…
Initializing variogram model…
Using 'hole-effect' Variogram Model
Partial Sill: 0.01269539252307871
Full Sill: 0.01825956409544843
Range: 1511.5351619286578
Nugget: 0.005564171572369719
```



Calculating statistics on variogram model fit…

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[3], line 1
----> 1 ok3d_hydr = compose_krige3D("Hydrate Saturation", 0.3, 'hole-effect')

Cell In[1], line 37, in compose_krige3D(name, sample_frac, variogram)
     34 values = sampled_data[name].values
     36 # Create the 3D Kriging model
---> 37 ok3d = OrdinaryKriging3D(
     38     X,
     39     Y,
     40     Z,
     41     values,
     42␣
  ↪      variogram_model=variogram,  # You might need to experiment with different models
     43     verbose=True,
     44     enable_plotting=True,
     45 )
     47 return ok3d
```

```
File ~/model-2024c/.venv/lib/python3.12/site-packages/pykrige/ok3d.py:352, in␣
 ↪OrdinaryKriging3D.__init__(self, x, y, z, val, variogram_model,␣
 ↪variogram_parameters, variogram_function, nlags, weight, anisotropy_scaling_y␣
 ↪anisotropy_scaling_z, anisotropy_angle_x, anisotropy_angle_y,␣
 ↪anisotropy_angle_z, verbose, enable_plotting, exact_values, pseudo_inv,␣
 ↪pseudo_inv_type)
    350 if self.verbose:
    351     print("Calculating statistics on variogram model fit…")
--> 352 self.delta, self.sigma, self.epsilon = _find_statistics(
    353     np.vstack((self.X_ADJUSTED, self.Y_ADJUSTED, self.Z_ADJUSTED)).T,
    354     self.VALUES,
    355     self.variogram_function,
    356     self.variogram_model_parameters,
    357     "euclidean",
    358     self.pseudo_inv,
    359 )
    360 self.Q1 = core.calcQ1(self.epsilon)
    361 self.Q2 = core.calcQ2(self.epsilon)

File ~/model-2024c/.venv/lib/python3.12/site-packages/pykrige/core.py:807, in␣
 ↪_find_statistics(X, y, variogram_function, variogram_model_parameters,␣
 ↪coordinates_type, pseudo_inv)
    804     continue
    806 else:
--> 807     k, ss = _krige(
    808         X[:i, :],
    809         y[:i],
    810         X[i, :],
    811         variogram_function,
    812         variogram_model_parameters,
    813         coordinates_type,
    814         pseudo_inv,
    815     )
    817     # if the estimation error is zero, it's probably because
    818     # the evaluation point X[i, :] is really close to one of the
    819     # kriging system points in X[:i, :]…
    820     # in the case of zero estimation error, the results are not stored
    821     if np.absolute(ss) < eps:

File ~/model-2024c/.venv/lib/python3.12/site-packages/pykrige/core.py:750, in␣
 ↪_krige(X, y, coords, variogram_function, variogram_model_parameters,␣
 ↪coordinates_type, pseudo_inv)
    748     res = np.linalg.lstsq(a, b, rcond=None)[0]
    749 else:
--> 750     res = np.linalg.solve(a, b)
    751 zinterp = np.sum(res[:n, 0] * y)
    752 sigmasq = np.sum(res[:, 0] * -b[:, 0])
```

```
File ~/model-2024c/.venv/lib/python3.12/site-packages/numpy/linalg/linalg.py:
 ↪409, in solve(a, b)
    407 signature = 'DD->D' if isComplexType(t) else 'dd->d'
    408 extobj = get_linalg_error_extobj(_raise_linalgerror_singular)
--> 409 r = gufunc(a, b, signature=signature, extobj=extobj)
    411 return wrap(r.astype(result_t, copy=False))

KeyboardInterrupt:
```