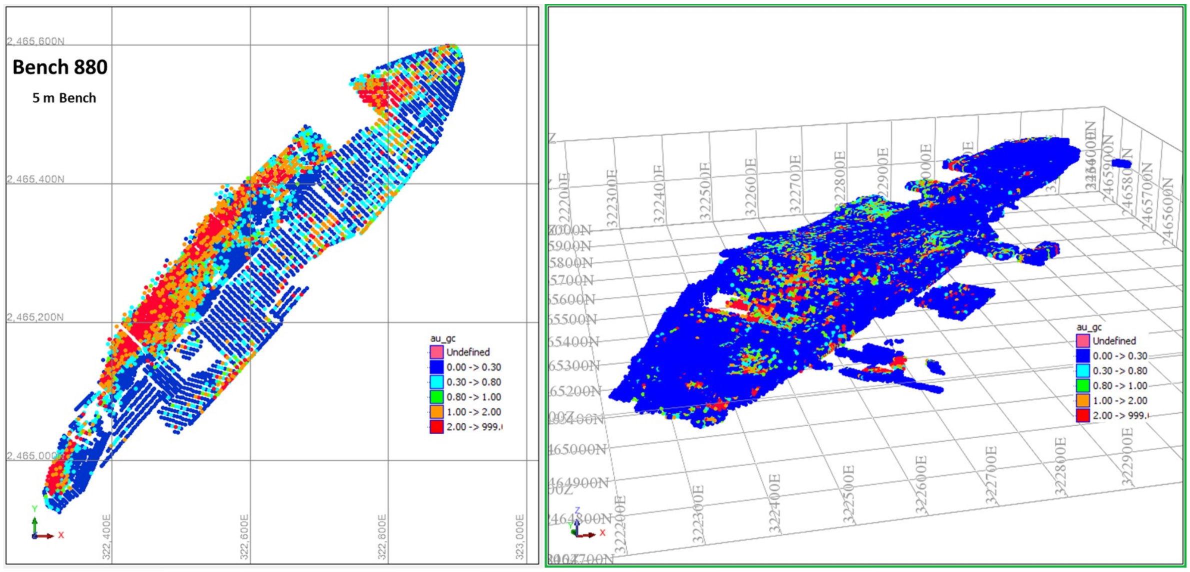
CPAG Project ,

Ordinary Kriging Estimation for Bench 880 and Compare The Result to EType Simulation Model

Data Used in This Project is from Ad Duwayhi Mine Located in Central of Saudi Arabia they Use Blasthole Samples for Grade Control @ 5 meter composite Samples



In [2]:
```
import pandas as pd
import numpy as np
import pygeostat as gs
import matplotlib
import matplotlib.pyplot as plt
import matplotlib.pylab as pylab
from IPython.display import display, Math, Latex
from IPython.display import Image
from DisplayPostscriptInJupyter import *
import warnings
import math
from tqdm.notebook import tqdm_notebook
from time import sleep
from tqdm import tqdm
import os
import sys
import seaborn as sns
import matplotlib as mpl
import cmocean
import time

warnings.filterwarnings('ignore')
%autosave 60
%matplotlib inline

print (' ')
print ('The versions of the python packages are given as follows:')
print (' ')
print ('Numpy version     :', np.__version__)
print ('Pandas version    :', pd.__version__)
print ('Pygeostat version :', gs.__version__)
print ('Python version    :', sys.version_info)
```

```
Autosaving every 60 seconds

The versions of the python packages are given as follows:

Numpy version     : 1.23.5
Pandas version    : 1.5.3
Pygeostat version : 1.1.1
Python version    : sys.version_info(major=3, minor=10, micro=9, releaselevel
='final', serial=0)
```

In [3]:
```
pd.set_option ('display.max_columns', 700)
pd.set_option ('display.max_rows', 400)
pd.set_option ('display.min_rows', 10)
pd.set_option ('display.expand_frame_repr', True)
```

In [4]:
```
plt.rcParams ['figure.figsize'] = (5.0, 5.0)
plt.style.use('seaborn-dark-palette')
plt.rcParams ['axes.grid'] = True
plt.rcParams ["patch.force_edgecolor"] = True
sns.set()
```

In [5]:
```
gs.Parameters['config.getpar'] = True
gs.Parameters['plotting.sigfigs'] = 2
gs.Parameters['plotting.histogram_plot.histbins'] = 20
gs.PlotStyle ['xtick.labelsize'] = 12
gs.PlotStyle ['ytick.labelsize'] = 12
gs.PlotStyle ['axes.labelsize'] = 11
```

In [6]:
```
print ('----------------------------------------------------------- ')
print ('The working directory for this study is given as follows:')
print ('------------------------------------------------------------ ')
print (' ')
print (os.getcwd ())
```

```
----------------------------------------------------------
The working directory for this study is given as follows:
-----------------------------------------------------------

C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE
```

In [7]:
```
os.chdir(r'C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnou
print ('-------------------------------------------------------------- ')
print ('The new working directory for this study is given as follows:')
print ('-------------------------------------------------------------- ')
print (' ')
print (os.getcwd ())
```

```
-------------------------------------------------------------
The new working directory for this study is given as follows:
-------------------------------------------------------------

C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE
```

In [8]:
```
datadir = 'datafiles/'
exedir = './exes/'
outdir = 'output/'
gs.mkdir = (outdir)
```

In [9]:
```
b_880 = gs.DataFile (datadir +'gold_b880.csv', dh= 'bh', x = 'x', y = 'y', z =
b_880.head(10)
```

Out[9]:

	bh	x	y	z	au
0	1.0	322451.28	2465152.40	877.5	0.084
1	2.0	322453.46	2465150.29	877.5	0.103
2	3.0	322455.54	2465148.16	877.5	0.044
3	4.0	322457.66	2465146.03	877.5	0.096
4	5.0	322406.75	2465090.89	877.5	0.321
5	6.0	322440.37	2465156.00	877.5	0.240
6	7.0	322441.40	2465155.24	877.5	0.323
7	8.0	322443.55	2465153.14	877.5	0.045
8	9.0	322445.68	2465151.01	877.5	0.038
9	10.0	322447.82	2465148.89	877.5	0.020

In [10]:
```
gs.write_gslib (b_880 ,datadir + 'b_880.out')
```

In [11]:
```
b880 = gs.DataFile (datadir + 'b_880.out',dh= 'bh', x = 'x', y = 'y', z = 'z')
```

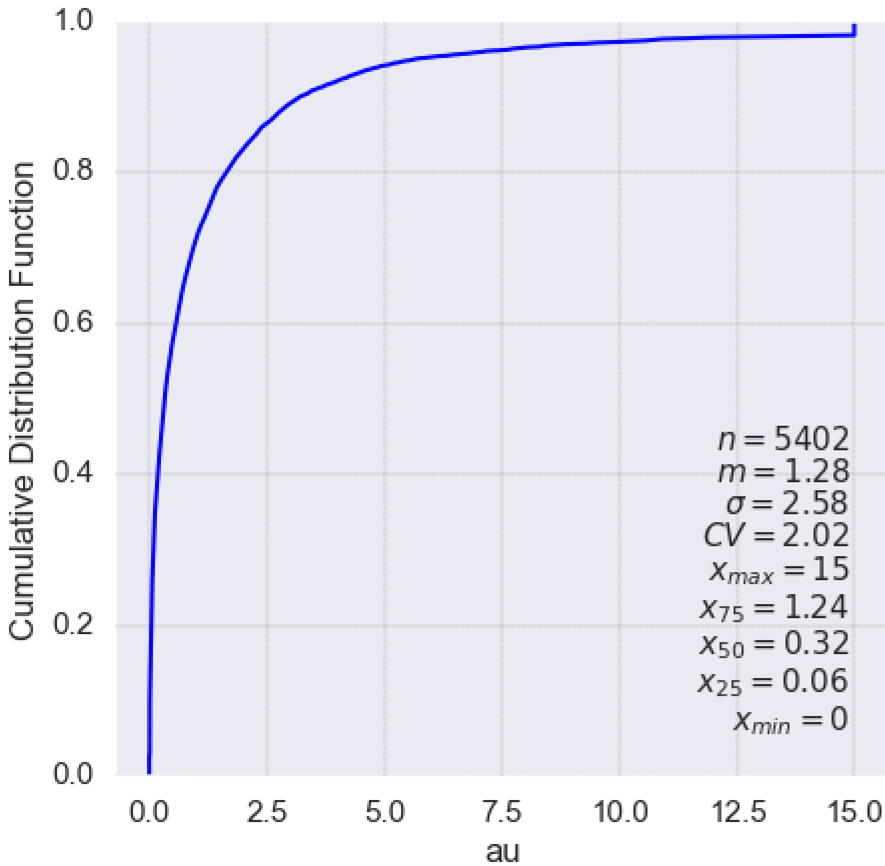
In [12]:
```
b880.head()
```

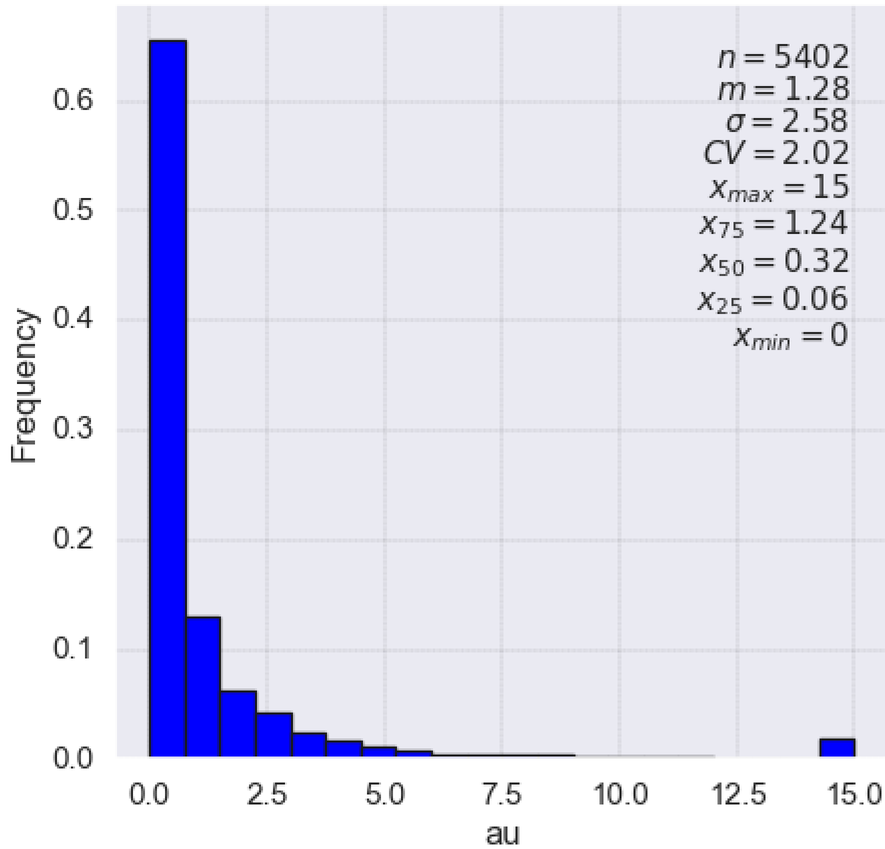
Out[12]:

	bh	x	y	z	au
0	1.0	322451.28	2465152.40	877.5	0.084
1	2.0	322453.46	2465150.29	877.5	0.103
2	3.0	322455.54	2465148.16	877.5	0.044
3	4.0	322457.66	2465146.03	877.5	0.096
4	5.0	322406.75	2465090.89	877.5	0.321

In [13]:
```
gs.histogram_plot (b880, var = 'au',
                    icdf = True, color = 'blue',stat_fontsize = 11, grid = Tr

gs.histogram_plot (b880, var = 'au',
                    icdf = False, color = 'blue',stat_fontsize = 11, grid = Tr
```

Out[13]: <Axes: xlabel='au', ylabel='Frequency'>



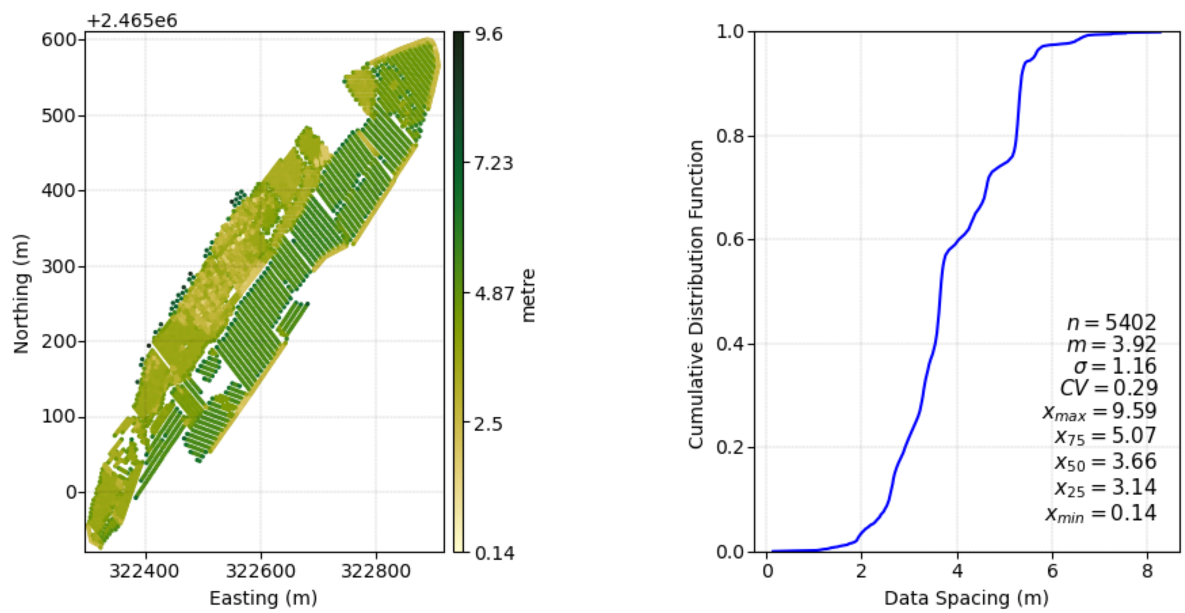


In [14]:
```
blksize = (10, 10, 5)
griddef_b880 = b880.infergriddef (blksize)
griddef_b880
```

Out[14]:
```
Pygeostat GridDef:
62 322301.0 10.0
69 2464926.0 10.0
3 874.5 5.0
```

In [15]:
```
mpl.style.use ('default')

b880.spacing (3)
b880 ['Data Spacing (m)'].describe ()

fig, axes = plt.subplots (1, 2, figsize = (9, 4.5))
gs.location_plot (b880, var = 'Data Spacing (m)',
                      cbar_label = 'metre', cmap = 'cmo.speed',
                      ax = axes [0], aspect = 1.3, s = 2,
                      grid = True, title = '', plot_style = True)

gs.histogram_plot (b880, var = 'Data Spacing (m)',
                      icdf = True, ax = axes [1], color = 'blue',stat_fontsize

fig.tight_layout ()
plt.subplots_adjust (left = 0.125, bottom = 0.1, right = 1.05, top = 0.95, wsp
plt.savefig (outdir + 'figures/Figure_1.png', bbox_inches = 'tight', dpi = 150
```

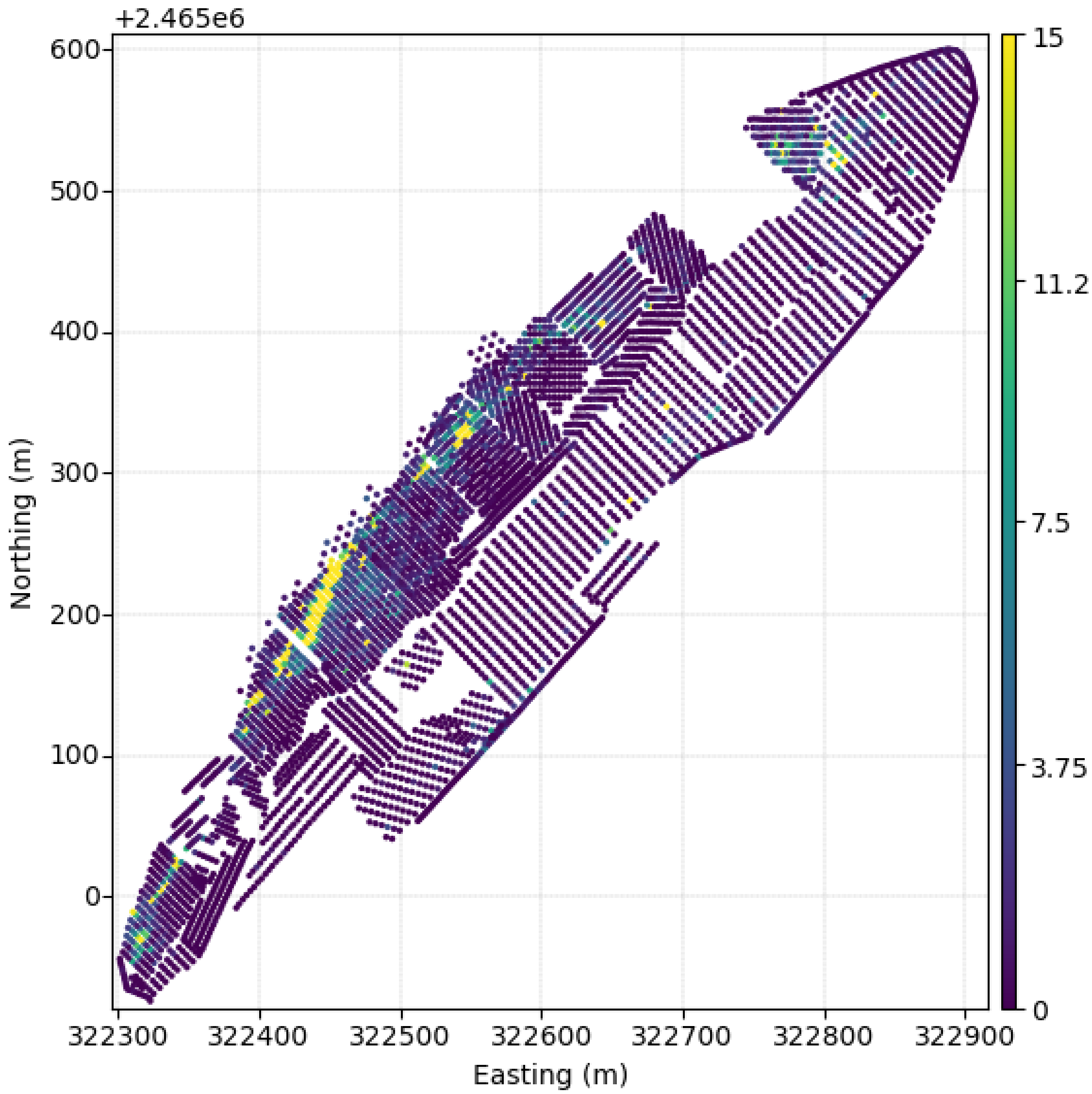
```
WARNING: current implementation of function is likely too memory intensivefor
greater than 5000 data
```



In [16]:
```
b880.spacing
```

Out[16]:
```
<bound method DataFile.spacing of DataFile: datafiles/b_880.out
Attributes:
dh: 'bh',  x: 'x',  y: 'y',  z: 'z',
Variables:
au
Grid Definitions:
62 322301.0 10.0
69 2464926.0 10.0
3 874.5 5.0>
```

In [17]:
```
mpl.style.use ('default')
gs.location_plot (b880, var = 'au', xlabel = 'Easting (m)', ylabel = 'Northing
                  figsize = (6, 10), s = 2, cmap = 'viridis', axis_xy = False,
                 cbar_label = 'Au g/t',grid = True, vlim = (0, 15), slice_numb
                 slicetol = 1, orient = 'xy', plot_style = True)

plt.savefig ('location_map.png', bbox_inches = 'tight', dpi = 150)
```



Declustring

Using declus.exe GSLIB program

In [18]:
```
declus = gs.Program (program = exedir + 'declus.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpx2ecevp5\declus.par has been copied to the clipboard
```

In [19]:
```
decluspar = """   Parameters for DECLUS
                   *********************

START OF PARAMETERS:
{datafl}                          - file with data
2   3   4   5                     - columns for X, Y, Z, and variable
-1.0e21     1.0e21                - trimming limits
{output_1}                        - file for summary output
{output_2}                        - file for output with data & weights
1.0   0.1                         - Y and Z cell anisotropy (Ysize=size*Yanis)
0                                 - 0=look for minimum declustered mean (1=max
{num}  {cmin}  {cmax}             - number of cell sizes, min size, max size
30                                - number of origin offsets
"""

declus.run (parstr = decluspar.format(datafl = datadir + 'b_880.out',
                                output_1 = outdir + './declus/b880_declus.sum'
                                output_2 = outdir + './declus/b880_declus.out'
            liveoutput = False)
print (' ')
print ('The two outputs of the declus program are given as follows: ')

declus_sum_b880 = gs.DataFile (outdir + './declus/b880_declus.sum', readfl = Tı
display (declus_sum_b880.head (n = 5))

declus_out_b880 = gs.DataFile (outdir + './declus/b880_declus.out', readfl = Tı
display (declus_out_b880.head (n = 5))
```

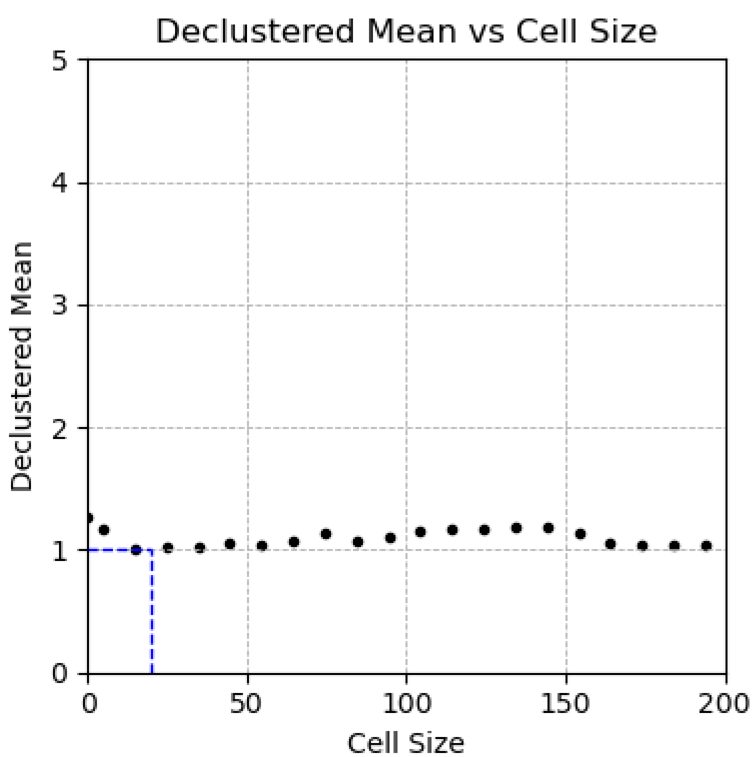
```
Calling:  ['./exes/declus.exe', 'temp']

The two outputs of the declus program are given as follows:
```

	Cell Size	Declustered Mean
0	0.000	1.276
1	5.000	1.173
2	14.938	1.013
3	24.875	1.016
4	34.812	1.029

	bh	x	y	z	au	Declustering Weight
0	1.0	322451.28	2465152.40	877.5	0.084	0.610346
1	2.0	322453.46	2465150.29	877.5	0.103	0.636794
2	3.0	322455.54	2465148.16	877.5	0.044	0.690385
3	4.0	322457.66	2465146.03	877.5	0.096	0.788164
4	5.0	322406.75	2465090.89	877.5	0.321	1.024121

Plot Cell Size

In [20]:
```
mpl.style.use ('default')

fig, ax1 = plt.subplots (1, 1, figsize = (4, 4))

ax1.scatter (declus_sum_b880 ['Cell Size'], declus_sum_b880 ['Declustered Mean
ax1.set_title ('Declustered Mean vs Cell Size')
ax1.set_xlabel ('Cell Size')
ax1.set_ylabel ('Declustered Mean')
ax1.set_xlim (0, 200)
ax1.set_ylim (0, 5)
ax1.plot ([20, 20], [0, 1], 'k--', lw = 1, color = 'blue')
ax1.plot ([0, 20], [1, 1], 'k--', lw = 1, color = 'blue')
plt.grid (linestyle = '--', linewidth = 0.6)

plt.tight_layout ()
plt.savefig (outdir + './figures/figure_4.png', bbox_inches = 'tight', dpi = 1
plt.show ()

print (' ')
print ('The data is observed at 20 m on the x-axis and 1.0 on the y-axis for ce
print (' ')
print ('The recommended cell size has been considered to be 20 x 20 m.')
print (' ')
```




```
The data is observed at 20 m on the x-axis and 1.0 on the y-axis for cell dec
lustring.

The recommended cell size has been considered to be 20 x 20 m.
```

Run DECLUS.EXE Program to determine cell size

In [20]:
```
declus = gs.Program (program = exedir + 'declus.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmp1fvv0d_n\declus.par has been copied to the clipboard
```

In [21]:
```
decluspar = """   Parameters for DECLUS
                   *********************

START OF PARAMETERS:
{datafl}                           - file with data
2   3   4   5                      - columns for X, Y, Z, and variable
-1.0e21     1.0e21                 - trimming limits
{output_1}                         - file for summary output
{output_2}                         - file for output with data & weights
1.0   0.1                          - Y and Z cell anisotropy (Ysize=size*Yanis)
0                                  - 0=look for minimum declustered mean (1=max
{num}  {cmin}  {cmax}              - number of cell sizes, min size, max size
25                                 - number of origin offsets
"""

declus.run (parstr = decluspar.format(datafl = datadir + 'b_880.out',
                                output_1 = outdir + './declus/b880_declus.sum'
                                output_2 = outdir + './declus/b880_declus.out'
            liveoutput = False)
print (' ')
print ('The two outputs of the declus program are given as follows: ')

declus_sum_b880 = gs.DataFile (outdir + './declus/b880_declus.sum', readfl = Tı
display (declus_sum_b880.head (n = 5))

declus_out_b880 = gs.DataFile (outdir + './declus/b880_declus.out', readfl = Tı
display (declus_out_b880.head (n = 5))
```

```
Calling:  ['./exes/declus.exe', 'temp']

The two outputs of the declus program are given as follows:
```

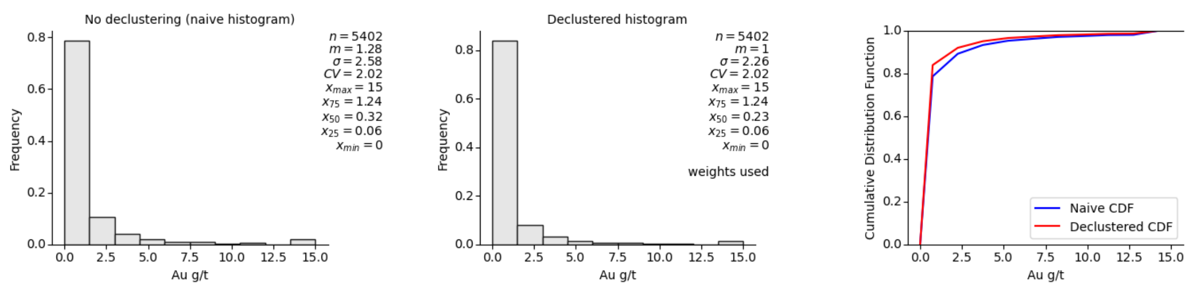
	Cell Size	Declustered Mean
0	0.0	1.276
1	20.0	1.002
2	20.0	1.002

	bh	x	y	z	au	Declustering Weight
0	1.0	322451.28	2465152.40	877.5	0.084	0.607697
1	2.0	322453.46	2465150.29	877.5	0.103	0.643587
2	3.0	322455.54	2465148.16	877.5	0.044	0.682181
3	4.0	322457.66	2465146.03	877.5	0.096	0.732994
4	5.0	322406.75	2465090.89	877.5	0.321	0.892190

Decluster Data to CSV

In [22]:
```
declus_out_b880.data.to_csv ('./output/declus/declus_b880.csv', index = False)
```

In [23]:
```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 3, figsize = (7, 3))
axes = axes.flatten ()

gs.histogram_plot (b880, var = 'au', bins = 10,
                      xlabel = 'Au g/t', grid = False, axis_xy = True,
                      stat_xy = (1.2, 1), ax = axes [0],
                      title = 'No declustering (naive histogram)',
                      plot_style = True)

gs.histogram_plot (declus_out_b880, var = 'au', ax = axes [1],
                      bins = 10, xlabel = 'Au g/t', grid = False,
                      axis_xy = True, weights = 'Declustering Weight',
                      stat_xy = (1.05, 1), title = 'Declustered histogram',
                      plot_style = True)

gs.histogram_plot (b880, var = 'au', bins = 10,
                      xlabel = 'Au g/t', icdf = True, color = 'blue',
                      grid = False, axis_xy = True, stat_blk = False,
                      ax = axes [2], label = 'Naive', plot_style = True)

gs.histogram_plot (declus_out_b880, var = 'au', ax = axes [2],
                      bins = 10, xlabel = 'Au g/t', grid = False,
                      axis_xy = True, weights = declus_out_b880.weights,
                      icdf = True, color = 'red', stat_blk = False,
                      label = 'Declustered', plot_style = True)

axes[2].legend(['Naive CDF', 'Declustered CDF'], loc = 4)
fig.tight_layout ()
plt.subplots_adjust (left = 0.125, bottom = 0.1, right = 1.95, top = 0.9, wspa

plt.savefig (outdir + 'figures/decl_Stat.png', bbox_inches = 'tight', dpi = 15
```



Normal Score Transform for all Data

In [24]:
```
nscoremv = gs.Program (program = exedir + 'nscoremv.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpn8qzxryz\nscoremv.par has been copied to the clipboard
```

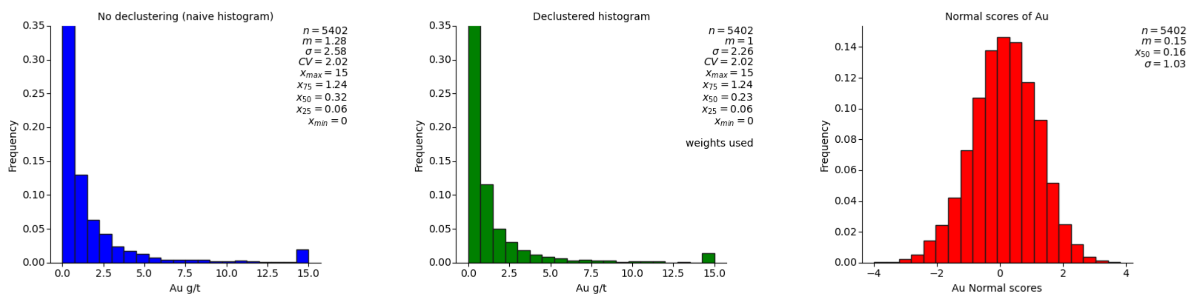
In [25]:
```
nscoremvpar = """Parameters for NSCOREMV
                ***********************

START OF PARAMETERS:
{datafl}                    -  file with data
1                           -  number of variables to transform
5                          -  columns for variables
6  0  0  0                  -  columns for weights
-1.0e21 1.0e21              -  trimming limits
{output}                    -  file for output
{first_trans}               -  file for first transformation table
0  0  0                     -  transform according to ref. dist., column numbe
nofile                      -  file with reference dist.
0  0  0                     -  transform according to ref. dist., column numbe
nofile                      -  file with reference dist.


**Notes:

To transform according to a reference distribution, set the first number to the
the second number to the column of the reference distribution variable, and the
to the weight if any.  If the variable column number is <= 0, this option will
considered so be sure to set it appropriately.
"""

nscoremv.run (parstr = nscoremvpar.format (datafl = outdir + './declus/b880_de
                                            output = outdir + './nscore/nscore
                                          first_trans =  outdir + './nscore/ns
```

```
Calling:  ['./exes/nscoremv.exe', 'temp']
```

In [27]:
```
b880_ns = gs.DataFile (outdir + '/nscore/nscore.out', x = 'x',
                         y = 'y', z = 'z',
                       dh = 'bh', griddef = griddef_b880,
                         readfl = True)
b880_ns.rename (columns = {'NScore: au':'NS_Au'})

gs.write_gslib (b880_ns, outdir + '/nscore/nscore_au.out')

display (b880_ns.head (n = 5))
```

	bh	x	y	z	au	Declustering Weight	NS_Au
0	1.0	322451.28	2465152.40	877.5	0.084	0.607697	-0.398536
1	2.0	322453.46	2465150.29	877.5	0.103	0.643587	-0.312614
2	3.0	322455.54	2465148.16	877.5	0.044	0.682181	-0.655091
3	4.0	322457.66	2465146.03	877.5	0.096	0.732994	-0.336659
4	5.0	322406.75	2465090.89	877.5	0.321	0.892190	0.158108

In [28]:
```
mpl.style.use ('default')

fig, axes = plt.subplots (1, 3, figsize = (8, 4))
axes = axes.flatten ()

gs.histogram_plot (b880, var = 'au', bins = 20, xlabel = 'Au g/t',color ='Blue
                   grid = False, axis_xy = True, stat_xy = (1.1, 1),
                   ax = axes [0], ylim = (0, 0.35),
                  title = 'No declustering (naive histogram)',
                   plot_style = True)
gs.histogram_plot (declus_out_b880, var = 'au', ax = axes [1],color ='Green',
                   bins = 20, xlabel = 'Au g/t',
                       grid = False, axis_xy = True,
                   weights = 'Declustering Weight',
                   stat_xy = (1.1, 1), ylim = (0, 0.35),
                  title = 'Declustered histogram',
                   plot_style = True)
gs.histogram_plot (b880_ns, var = 'NS_Au', bins = 20,color ='Red',
                   xlabel = 'Au Normal scores', icdf = False,
                   grid = False, axis_xy = True,
                   stat_xy = (1.2, 1), ax = axes [2],
                  title = 'Normal scores of Au',
                   stat_blk = 'minimal', plot_style = True)

print (' ')
fig.tight_layout ()
plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 1.95, top = 0.9, wspace = 0.5,
                     hspace = 0.35)
plt.savefig (outdir + 'figures/dec_NS_st.png',
             bbox_inches = 'tight', dpi = 150)
```



Dcluse and NScore for The lime Domain

Laod the `varmap.exe` `GSLIB` program and its parameter file

In [29]:
```
varmap = gs.Program (program = exedir + 'varmap.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmplknpuhpc\varmap.par has been copied to the clipboard
```

In [30]:
```
varmappar = """   Parameters for VARMAP
                  *********************

START OF PARAMETERS:
./output/nscore/nscore_au.out                 - file with data
1   7                                    - number of variables: column numbers
-1.0e21     1.0e21                       - trimming limits
0                                        - 1=regular grid, 0=scattered values
50   50    50                            - if =1: nx,     ny,   nz
25.0  25.0  5                          - xsiz, ysiz, zsiz
2   3   4                                - if =0: columns for x,y, z coordinate
./output/varmap/varmap_b880_ns.out     - file for variogram output
 9    9     9                            - nxlag, nylag, nzlag
50.0   50.0   1.0                      - dxlag, dylag, dzlag
5                                        - minimum number of pairs
1                                        - standardize sill? (0=no, 1=yes)
1                                        - number of variograms
1   1   1                                - tail, head, variogram type


type 1 = traditional semivariogram
     2 = traditional cross semivariogram
     3 = covariance
     4 = correlogram
     5 = general relative semivariogram
     6 = pairwise relative semivariogram
     7 = semivariogram of logarithms
     8 = semimadogram
     9 = indicator semivariogram - continuous
     10= indicator semivariogram - categorical
"""

varmap.run (parstr = varmappar, liveoutput = False)
```

```
Calling:  ['./exes/varmap.exe', 'temp']
```

In [31]:
```
griddef_3 = gs.GridDef (grid_file = 'varmap_grid_b880.txt')
griddef_3
```

Out[31]:
```
Pygeostat GridDef:
21 -10.0 1.0
21 -10.0 1.0
21 -10.0 1.0
```

In [32]:
```
varmap_b880_ns = gs.DataFile (outdir + './varmap/varmap_b880_ns.out',
                                griddef = griddef_3)
varmap_b880_ns.head (n=5)
```

Out[32]:

	variogram	number of pairs	head mean	tail mean	head variance	tail variance
0	NaN	0.0	NaN	NaN	NaN	NaN
1	NaN	0.0	NaN	NaN	NaN	NaN
2	NaN	0.0	NaN	NaN	NaN	NaN
3	NaN	0.0	NaN	NaN	NaN	NaN
4	NaN	0.0	NaN	NaN	NaN	NaN

I Face an isssue ploting the Variogram map

Experimental Variogram calcuation for the Normal Score Data

In [34]:
```
varcalc = gs.Program (program = exedir + 'varcalc', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmp05dv7byq\varcalc.par has been copied to the clipboard
```

In [35]:
```
varcalcpar = """  Parameters for VARCALC
                  **********************

START OF PARAMETERS:
./output/nscore/nscore_au.out                         - file with data
2   3   4                                           - columns for X, Y, Z coordina
1   7   0                                           - number of variables,column nu
-998.0    1.0e21                                    - trimming limits
3                                                   - number of directions
45  22.5 9999 0.0 0.0 5.0 0.0                     - Major: azm,azmtol,bandhorz,dip
30  10  35                                        - number of lags,lag distance,la
135.0 22.5 999 0.0 0.0 5.0 0.0                     - Minor: azm,azmtol,bandhorz,di
30  10  35                                         - number of lags,lag distance,l
0.0 0.0 999 -90.0 0.0 5.0 0.0                     - Minor: azm,azmtol,bandhorz,dip
30 10   5                                        - number of lags,lag distance,lag
./output/experimental/exp2_var_b880_ns.out         - file for experimental variogr
0                                                   - legacy output (0=no, 1=write
1                                                   - run checks for common errors
1                                                   - standardize sills? (0=no, 1=
1                                                   - number of variogram types
1   1   1   ?                                       - tail variable, head variable




NOTES ON VARIOGRAM CALCULATION:
1) By default, varcalc runs checks for common errors in parameter choices. Thi
   disabled if desired.
2) Varcalc can standardize using a provided sill (such as a declustered varian
   For example, if variable 1 has a declustered variance of 8.6, the tradition
   semivariogram could be standardized by setting the variogram type to:
      1  1  1  8.6
   Alternatively, varcalc can attempt to infer a sill for standardizing by set
   the variogram type to:
      1  1  1  ?
   The calculated sills will be written to the console.
3) Variogram types are the same as in GSLIB:
      1 = traditional semivariogram
      2 = traditional cross semivariogram
      3 = covariance   (-3 calculates variance (provided sill) -covariance)
      4 = correlogram  (-4 calculates 1-correlation)
      5 = general relative semivariogram
      6 = pairwise relative semivariogram
      7 = semivariogram of logarithms
      8 = semimadogram
      9 = indicator semivariogram - continuous  - requires a cutoff
      10= indicator semivariogram - categorical - requires a category
4) For indicator variograms, the variogram cutoff/categories are specified imm
    1   1   9   1.0   ?                -tail variable, head variable, variogram
5) If desired, the program can write out the variogram points in the gamv2004 
   for compatibility with older versions. Tilt was not supported in pre-varcal
   programs so use carefully.
"""

varcalc.run (parstr = varcalcpar, liveoutput = False)
```

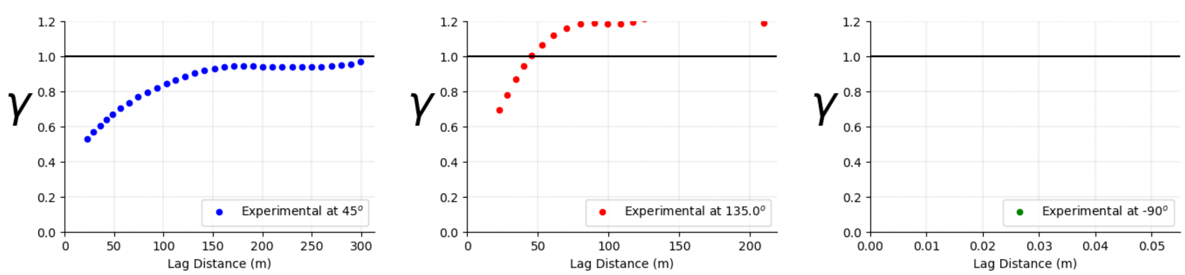
```
Calling:  ['./exes/varcalc', 'temp']
```

In [36]:
```
exp_var_b880_ns = gs.DataFile (outdir + 'experimental/exp2_var_b880_ns.out')
exp_var_b880_ns.head ()
```

Out[36]:

	Variogram Index	Lag Distance	Number of Pairs	Variogram Value	Variogram Number	Calculation Azimuth	Calculation Dip	Variogram Type
0	1.0	23.019419	125321.0	0.528038	1.0	45.0	0.0	1.0
1	1.0	29.470508	201439.0	0.569062	1.0	45.0	0.0	1.0
2	1.0	35.795068	291123.0	0.606018	1.0	45.0	0.0	1.0
3	1.0	42.280881	397593.0	0.639856	1.0	45.0	0.0	1.0
4	1.0	48.869169	513343.0	0.670084	1.0	45.0	0.0	1.0

In [37]:
```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 3, figsize = (7, 3))
axes = axes.flatten ()

lab = ['Experimental at $45^{o}$', 'Experimental at $135.0^{o}$', 'Experimenta
colors = ['blue', 'red', 'green']

for i, ax, j, m in zip (lab, axes, range (3), colors):
    j = j + 1
    gs.variogram_plot (exp_var_b880_ns.data, index = j,
                       grid = True,
                       color = m, minpairs = True, label = i,
                       experimental = True,
                        ax = ax, axis_xy = True, unit = 'm',
                       ms = 5.5, pairnumbers = False)
    ax.legend (loc = 4, fontsize = 10)
    ax.xaxis.label.set_size (10)

fig.tight_layout ()

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 1.95, top = 0.9, wspace = 0.3,
                     hspace = 0.55)

plt.savefig (outdir + 'figures/ex_vaio_ns.png',
             bbox_inches = 'tight', dpi = 150)
```



In [38]:
```
varmodel = gs.Program (program = exedir + 'varmodel', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpgisqehiv\varmodel.par has been copied to the clipboard
```

In [39]:
```
varmodelpar = """ Parameters for VARMODEL
                  **********************

START OF PARAMETERS:
./output/model/varmodel_dirc_b880_ns.out   - file for modeled variogram points
3                                             - number of directions to model p
  45.0 0.0  5500   0.5                        - azm, dip, npoints, point separa
  135.0 0.0  5500   0.5                       - azm, dip, npoints, point separa
  0.0  90.0  180   0.5                        - azm, dip, npoints, point separa
3    0.0                                      - nst, nugget effect
1    0.50  45   0.0   0.0                    - it,cc,ang1,ang2,ang3
        25.0  49.0  0.01                    - a_hmax, a_hmin, a_vert
1    0.40  45   0.0   0.0                    - it,cc,ang1,ang2,ang3
       160.0 40.0 0.03                        - a_hmax, a_hmin, a_vert
1    0.10  45   0.0   0.0                   - it,cc,ang1,ang2,ang3
       1000.0 1.0  0.01                    - a_hmax, a_hmin, a_vert
0   100000                                    - fit model (0=no, 1=yes), maximu
1.0                                           - variogram sill (can be fit, but
1                                             - number of experimental files to
varcalc.out                                   - experimental output file 1
2   1   4                                     - # of variograms (<=0 for all),
1   1   10                                    - # pairs weighting, inverse dist
0     10.0                                    - fix Hmax/Vert anis. (0=no, 1=ye
0      1.0                                    - fix Hmin/Hmax anis. (0=no, 1=ye
varmodelfit.var                               - file to save fit variogram mode



NOTES ON VARIOGRAM FITTING:
1) This program can be run as the GSLIB program vmodel where an already
   fit variogram model is provided.
2) Alternatively, a variogram model can be fit. Any parameter, except
   the number of structures can be fit. Fitting variogram angles
   is NOT recommended best practice. Options for fitting are:
   ? - fit the parameter with no constraints
   a:b - fit the parameter between a and b
   a: - fit the parameter so it is >=a
   :b - fit the parameter so it is <=b
   There must be no spaces in a:b!
3) Structure types (it) are:
   1 - spherical variogram model
   2 - exponential variogram model
   3 - gaussian variogram model
   4 - hole effect variogram model (cannot be automatically fit)
"""
varmodel.run (parstr = varmodelpar, liveoutput = False)

varmodel_dirc_b880_ns = gs.DataFile (outdir +'model/varmodel_dirc_b880_ns.out'
varmodel_dirc_b880_ns.head ()
```

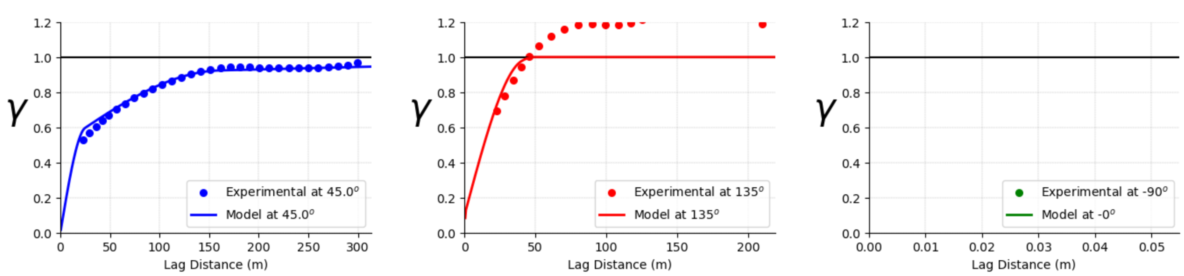
```
Calling:  ['./exes/varmodel', 'temp']
```

Out[39]:

	Variogram Index	Lag Distance	Number of Pairs	Variogram Value	Variogram Number	Calculation Azimuth	Calculation Dip
0	1.0	0.5	1.0	0.016948	1.0	45.0	0.0
1	1.0	1.0	1.0	0.033884	1.0	45.0	0.0
2	1.0	1.5	1.0	0.050796	1.0	45.0	0.0
3	1.0	2.0	1.0	0.067672	1.0	45.0	0.0
4	1.0	2.5	1.0	0.084499	1.0	45.0	0.0

In [40]:

```
mpl.style.use ('default')

fig, axes = plt.subplots (1, 3, figsize = (7, 3))

axes = axes.flatten ()
lab = ['Experimental at $45.0^{o}$', 'Experimental at $135^{o}$', 'Experimenta
lab_2 = ['Model at $45.0^{o}$', 'Model at $135^{o}$', 'Model at -$0^{o}$']
colors = ['blue', 'red', 'green']

for i, ax, j, k, m in zip (lab, axes, range (3), lab_2, colors):
    j = j + 1
    gs.variogram_plot (exp_var_b880_ns.data, index = j,  grid = True,
                       color = m, minpairs = False,
                       label = i, experimental = True,
                        ax = ax, axis_xy = True, unit = 'm',
                       ms = 5.5, plot_style = True)
    gs.variogram_plot (varmodel_dirc_b880_ns.data, index = j,
                       sill = False, grid = False, minpairs = False,
                        label = k, experimental = False, ax = ax,
                       axis_xy = True, unit = 'm'
                       , ms = 0, lw = 2, color = m, plot_style = True)
    ax.legend (loc = 4, fontsize = 10)
    ax.xaxis.label.set_size (10)

fig.tight_layout ()

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 1.95, top = 0.9, wspace = 0.3, hspace = 0.55)

plt.savefig (outdir + 'figures/vaimod.png',
             bbox_inches = 'tight', dpi = 150)
```



In [47]:
```
b880_vario_ns = '''
3    0.0                                      - nst, nugget effect
1    0.50  45   0.0   0.0                    - it,cc,ang1,ang2,ang3
        25.0  49.0  0.01                    - a_hmax, a_hmin, a_vert
1    0.40  45   0.0   0.0                    - it,cc,ang1,ang2,ang3
       160.0 40.0 0.03                        - a_hmax, a_hmin, a_vert
1    0.10  45   0.0   0.0                   - it,cc,ang1,ang2,ang3
       1000.0 1.0  0.01                    - a_hmax, a_hmin, a_vert
                  '''
print ('')
print ('The parameters of the analytical model fitted to the experimental vario
print ('')
print (b880_vario_ns)
```

```
The parameters of the analytical model fitted to the experimental variogram o
f the normal scores of the Au grade is:


3    0.0                                      - nst, nugget effect
1    0.50  45   0.0   0.0                    - it,cc,ang1,ang2,ang3
        25.0  49.0  0.01                    - a_hmax, a_hmin, a_vert
1    0.40  45   0.0   0.0                    - it,cc,ang1,ang2,ang3
       160.0 40.0 0.03                        - a_hmax, a_hmin, a_vert
1    0.10  45   0.0   0.0                   - it,cc,ang1,ang2,ang3
       1000.0 1.0  0.01                    - a_hmax, a_hmin, a_vert
```

Use sequential Gaussian simulation to simulate 200 realizations of the normal score variable

In [48]:
```
sgsim = gs.Program (program = exedir + 'sgsim.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmprw_czcs3\sgsim.par has been copied to the clipboard
```

In [49]:
```
sgsimpar = """    Parameters for SGSIMv4
                  *********************

START OF PARAMETERS:
{datafl}                             - file with data
2  3  4  7  0  0                     - columns for X,Y,Z,vr,wt,sec.var.
-998.0      1.0e21                   - trimming limits
0                                    - transform the data (0=no, 1=yes)
sgsim.trn                            - file for output trans table
0                                    - consider ref. dist (0=no, 1=yes)
histsmth.out                         - file with ref. dist distribution
1  2                                 - columns for vr and wt
4.0    11.5                          - zmin,zmax (for tail extrapolation)
1       4.0                          - lower tail option (1=linear), parameter
1      11.5                          - upper tail option (1=linear), parameter
1                                    - debugging level: 0,1,2,3
sgsim.dbg                            - file for debugging output
{output}                             - file for simulation output
{real}                               - number of realizations to generate
{grid}
56693                                - random number seed
0     40                             - min and max original data for sim
40                                   - number of simulated nodes to use
1                                    - assign data to nodes (0=no, 1=yes)
1     3                              - multiple grid search (0=no, 1=yes),num
0                                    - maximum data per octant (0=not used)
1000.0  1000.0  40.0                  - maximum search radii (hmax,hmin,vert)
45.0   0.0   0.0                     - angles for search ellipsoid
0     0.60   1.0                     - ktype: 0=SK,1=OK,2=LVM,3=EXDR,4=COLC, co
nofile                               - file with LVM, EXDR, or COLC variable
0                                    - column for secondary variable
{varmodel}
"""

pars = dict (datafl = outdir + '/nscore/nscore_au.out',
             output = outdir + '/simulation/sgsim.out',
                                     real = 10, grid = griddef_b880,
             varmodel = b880_vario_ns)
sgsim.run (parstr = sgsimpar.format (**pars), liveoutput = False)
```

```
Calling:  ['./exes/sgsim.exe', 'temp']
```

Load the simulated realizations

In [50]:
```
simu_real = gs.DataFile (outdir + '/simulation/sgsim.out',
                         griddef = griddef_b880)
simu_real.head ()
```

Out[50]:

	value
0	1.026144
1	0.565773
2	-0.355219
3	0.504868
4	1.064882

In [51]:
```
addcoord = gs.Program (program = exedir + 'addcoord.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpvm_njted\addcoord.par has been copied to the clipboard
```

In [52]:
```
addcoordpar = """                 Parameters for ADDCOORD
                  **********************


START OF PARAMETERS:
{datafl}                         -file with data
{output}                      -file for output
10                               -realization number
{grid}
3 3 6                            -decimals for x, y, z (-1=exclude)
"""
addcoord.run (parstr = addcoordpar.format (datafl = outdir+'simulation/sgsim.o
                                           output = outdir+'/simulation/coord.o
```

```
Calling:  ['./exes/addcoord.exe', 'temp']

 ADDCOORD VERSION: 4.1.0

 Data File = output/simulation/sgsim.out
 Output File = output//simulation/coord.out
 Realization number =          10
 X grid size =          62   322301.000000000        10.0000000000000
 Y grid size =          69   2464926.00000000        10.0000000000000
 Z grid size =           3   874.500000000000        5.00000000000000
 decimals =            3            3            6

 Format: (f10.03,x,f11.03,x,f10.06,x,a)

 ADDCOORD: 4.1.0 Finished
```

In [53]:
```
a = simu_real ['value']
for t, x in enumerate (a):
    print (t,x)
```

```
0 1.0261443
1 0.56577343
2 -0.35521913
3 0.50486779
4 1.0648819
5 -0.61865145
6 -1.3798223
7 -0.42816114
8 -1.6881193
9 -1.436774
10 -1.2008598
11 -1.3894863
12 -1.2954336
13 -0.28406721
14 0.071980089
15 1.7080758
16 0.38392189
17 -0.25170901
18 -0.45902455
```

In [54]:
```
a = simu_real ['value']

reals = []

for t, x in enumerate (a):
    if t < 11:
        plot_real = t + 1
        real_nodes = griddef_b880.nx * griddef_b880.ny * griddef_b880.nz
        i = real_nodes * (plot_real - 1)
        j = real_nodes * plot_real
        c = a [i:j]
        reals.append (c)
    else:
        break
print (reals)
```

```
[0        1.026144
1        0.565773
2       -0.355219
3        0.504868
4        1.064882
           ...
12829   -1.092246
12830    0.012751
12831    0.814712
12832    0.058679
12833   -0.630790
Name: value, Length: 12834, dtype: float64, 12834    0.739041
12835    0.703035
12836    0.082069
12837   -0.716915
12838    0.101387
           ...
25663   -0.316633
25664   -0.834554
```

convert the Reals into Datafram

In [55]:
```
cc = []

for i in range (11):
    ff = pd.DataFrame (reals[i])
    cc.append (ff)

for i, j in zip (range (11), range (11)):
    j = j + 1
    gs.write_gslib (cc [i], outdir + '/simulation/real_{}.out'.format (j))
```

In [56]:
```
cc [9]
```

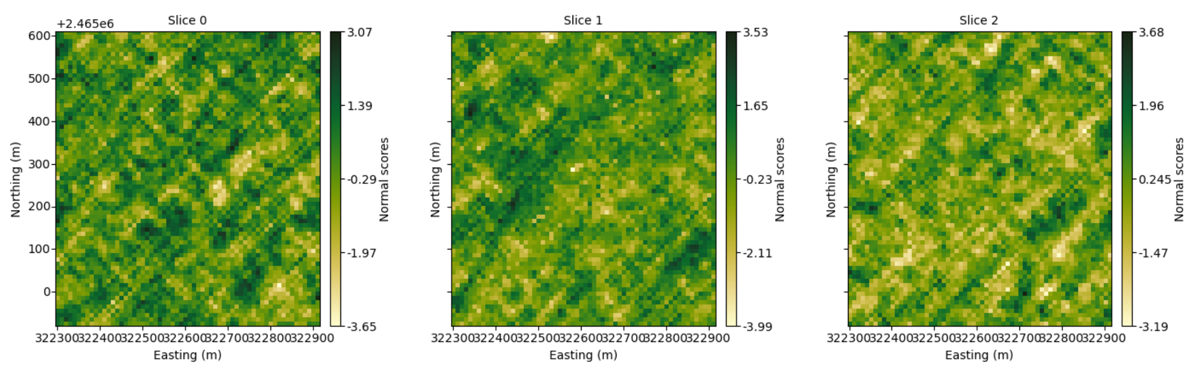
Out[56]:

	value
115506	-0.314029
115507	0.247717
115508	0.137873
115509	1.971948
115510	2.225748
...	...
128335	0.977994
128336	0.191041
128337	-0.539601
128338	-1.867563
128339	-0.665549

12834 rows × 1 columns

Display some sections from the first realizations

In [57]:
```
mpl.style.use ('default')
reals_two = []

for i in range (2):
    i = i + 1
    a = gs.DataFile (outdir + 'simulation/real_{}.out'.format (i), griddef = g
    reals_two.append (a)

fig, axes = plt.subplots (1, 3, figsize = (15, 15), sharey = True)

axes = axes.flatten ()

slices = []

for i in range (0, 3):
    slices.append (i)

for ax, i in zip (axes, slices):
    gs.slice_plot (reals_two [0], var = 'value',
                   orient = 'xy', cmap = 'cmo.speed',
                   title = 'Slice ' + str(i),
               cbar_label = 'Normal scores', grid = False,
                   ax = ax, unit = 'm', slice_number = i,
                   plot_style = True)

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 0.95, top = 0.55,
                     wspace = 0.50, hspace = 0.4)
plt.savefig (outdir + 'figures/figure_7.png',
             bbox_inches = 'tight', dpi = 150)
```



Calculate the E-type model

Load the `postsim.exe` `GSLIB` program and its parameter file

In [58]:
```
postsim = gs.Program (program = exedir + 'postsim.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmp0kxubg0t\postsim.par has been copied to the clipboard
```

In [59]:
```
postsimpar = """  Parameters for POSTSIM
                  **********************

START OF PARAMETERS:
{datafl}                             - file with realizations
10      1                           - number of realizations, optional column n
-1.0e21   1.0e21                     - trimming limits
{nx}  {ny}   {nz}                    - nx, ny, nz
{output}                             - file for output array(s)
1   0.0                              - output option, output parameter
3  10 11 12                          - if 7,  number of categories, categories
kt3d.out                             - if 10, kriged model to correct to


option 1 = E-type mean and conditional variance
       2 = prob and mean above threshold (par)
       3 = Z-percentile corresponding to (par)
       4 = symmetric (par) probability interval
       6 = prob to be within (par) % of the mean
       7 = summarize categorical variable realizations
      10 = correct to kriged model
"""

postsim.run (parstr = postsimpar.format (datafl = outdir + 'simulation/sgsim.ou
                                        nx = griddef_b880.nx, ny = griddef_b880
                                         nz = griddef_b880.nz,
                                        output = './output/simulation/etype.out
```

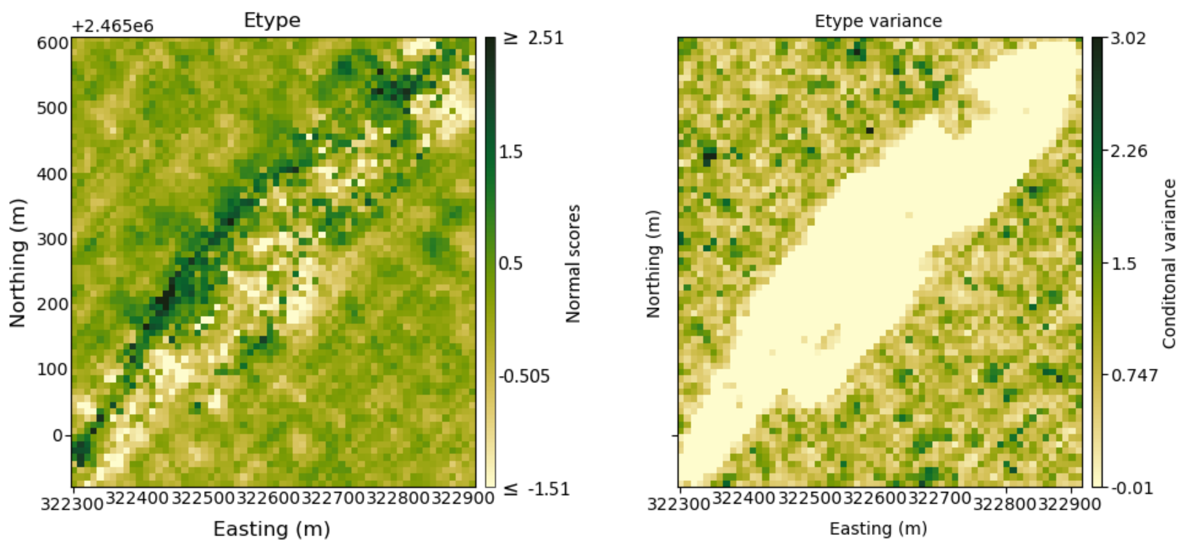
```
Calling:  ['./exes/postsim.exe', 'temp']
```

In [60]:
```
etype = gs.DataFile (outdir + '/simulation/etype.out', griddef = griddef_b880,
etype.head ()
```

Out[60]:

	E-type	variance
0	0.511252	0.501674
1	0.413135	0.339269
2	0.341838	0.401719
3	0.157759	1.814068
4	0.464288	0.873328

In [61]:
```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 2, figsize = (10, 10), sharey = True)
axes = axes.flatten ()

gs.slice_plot (etype, var = 'E-type', orient = 'xy',
               cmap = 'cmo.speed',  title = 'Etype',
               cbar_label = 'Normal scores',
               grid = False, ax = axes [0], unit = 'm',
                  slice_number = 1, vlim = (-1.51, 2.51))
gs.slice_plot (etype, var = 'variance', orient = 'xy',
               cmap = 'cmo.speed',  title = 'Etype variance',
               cbar_label = 'Conditonal variance',
               grid = False, ax = axes [1], unit = 'm',
                  plot_style = True, slice_number = 1)

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 0.95, top = 0.55, wspace = 0.50,
                     hspace = 0.4)
plt.savefig (outdir + 'figures/figure_9.png',
             bbox_inches = 'tight', dpi = 150)
```



In [62]:
```
sgsim = gs.Program (program = exedir + 'sgsim.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpdemqraez\sgsim.par has been copied to the clipboard
```

In [63]:
```
sgsimpar = """    Parameters for SGSIMv4
                  **********************

START OF PARAMETERS:
{datafl}                             - file with data
2  3  4  5  6  0                     - columns for X,Y,Z,vr,wt,sec.var.
-998.0       1.0e21                  - trimming limits
1                                    - transform the data (0=no, 1=yes)
sgsim.trn                            - file for output trans table
0                                    - consider ref. dist (0=no, 1=yes)
histsmth.out                         - file with ref. dist distribution
1  2                                 - columns for vr and wt
0.0    15.0                         - zmin,zmax (for tail extrapolation)
1       0.0                          - lower tail option (1=linear), parameter
1      16.66                         - upper tail option (1=linear), parameter
1                                    - debugging level: 0,1,2,3
sgsim.dbg                            - file for debugging output
{output}                             - file for simulation output
{real}                               - number of realizations to generate
{grid}
59673                                - random number seed
0     40                             - min and max original data for sim
40                                   - number of simulated nodes to use
1                                    - assign data to nodes (0=no, 1=yes)
1     3                              - multiple grid search (0=no, 1=yes),num
0                                    - maximum data per octant (0=not used)
1000.0  100.0  20.0                  - maximum search radii (hmax,hmin,vert)
 90.0   0.0   0.0                    - angles for search ellipsoid
0     0.60   1.0                     - ktype: 0=SK,1=OK,2=LVM,3=EXDR,4=COLC, co
nofile                               - file with LVM, EXDR, or COLC variable
1                                    - column for secondary variable
{varmodel}
"""

sgsim.run (parstr = sgsimpar.format (datafl = outdir + '/declus/b880_declus.ou
                                     output = outdir + '/simulation/sgsim_tr.o
                                    real = 10, grid = griddef_b880, varmodel =
           liveoutput = True)
```

```
Calling:  ['./exes/sgsim.exe', 'temp']

 SGSIM Version: 4.000

  data file = output//declus/b880_declus.out
  input columns =            2           3           4           5
6
           0
  trimming limits =   -998.0000      1.0000000E+21
  transformation flag =            1
  transformation file = sgsim.trn
  consider smoothed distribution (1=yes) =            0
  file with smoothed distribution = histsmth.out
  columns =            1           2
  data limits (tails) =   0.0000000E+00   15.00000
  lower tail =            1  0.0000000E+00
  upper tail =            1   16.66000
  debugging level =            1
  debugging file = sgsim.dbg
```

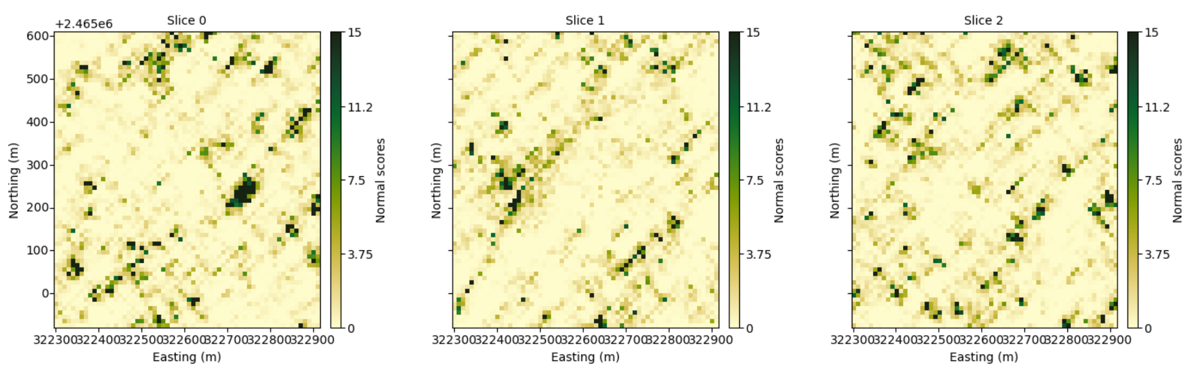
In [64]:

```
simu_real_tr = gs.DataFile (outdir + '/simulation/sgsim_tr.out', griddef = gri
print (simu_real_tr.head ())
```

```
      value
0  0.010000
1  0.384000
2  0.366304
3  0.401000
4  0.298559
```

In [65]:
```
a = simu_real_tr ['value']

reals_tr = []

for t, x in enumerate (a):
    if t < 201:
        plot_real = t + 1
        real_nodes = griddef_b880.nx * griddef_b880.ny * griddef_b880.nz
        i = real_nodes * (plot_real - 1)
        j = real_nodes * plot_real
        c = a [i:j]
        reals_tr.append (c)
    else:
        break

cc_tr = []

for i in range (11):
    ff = pd.DataFrame (reals_tr [i])
    cc_tr.append (ff)

for i, j in zip (range (11), range (11)):
    j = j + 1
    gs.write_gslib (cc_tr [i], outdir + '/simulation/real_tr_{}.out'.format (j
```

In [66]:
```
mpl.style.use ('default')
reals_two = []

for i in range (2):
    i = i + 1
    a = gs.DataFile (outdir + 'simulation/real_tr_{}.out'.format (i), griddef 
    reals_two.append (a)

fig, axes = plt.subplots (1, 3, figsize = (15, 15), sharey = True)

axes = axes.flatten ()

slices = []

for i in range (0, 3):
    slices.append (i)

for ax, i in zip (axes, slices):
    gs.slice_plot (reals_two [0], var = 'value',
                   orient = 'xy', cmap = 'cmo.speed',
                   title = 'Slice ' + str(i),
               cbar_label = 'Normal scores', grid = False,
                   ax = ax, unit = 'm', slice_number = i,
                   plot_style = True)

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 0.95, top = 0.55,
                     wspace = 0.50, hspace = 0.4)
plt.savefig (outdir + 'figures/figure_7.png',
             bbox_inches = 'tight', dpi = 150)
```



In [67]:
```
postsim = gs.Program (program = exedir + 'postsim.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpsw5ih5cz\postsim.par has been copied to the clipboard
```

In [68]:
```
postsimpar = """  Parameters for POSTSIM
                  *********************

START OF PARAMETERS:
{datafl}                             - file with realizations
10      1                           - number of realizations, optional column r
-1.0e21   1.0e21                     - trimming limits
{nx}  {ny}   {nz}                    - nx, ny, nz
{output}                             - file for output array(s)
1   0.0                              - output option, output parameter
3  10 11 12                          - if 7,  number of categories, categories
kt3d.out                             - if 10, kriged model to correct to


option 1 = E-type mean and conditional variance
       2 = prob and mean above threshold (par)
       3 = Z-percentile corresponding to (par)
       4 = symmetric (par) probability interval
       6 = prob to be within (par) % of the mean
       7 = summarize categorical variable realizations
      10 = correct to kriged model
"""

postsim.run (parstr = postsimpar.format (datafl = outdir + 'simulation/sgsim_tr
                                         nx = griddef_b880.nx, ny = griddef_b880
                                          nz = griddef_b880.nz,
                                         output = './output/simulation/etype_tr
```

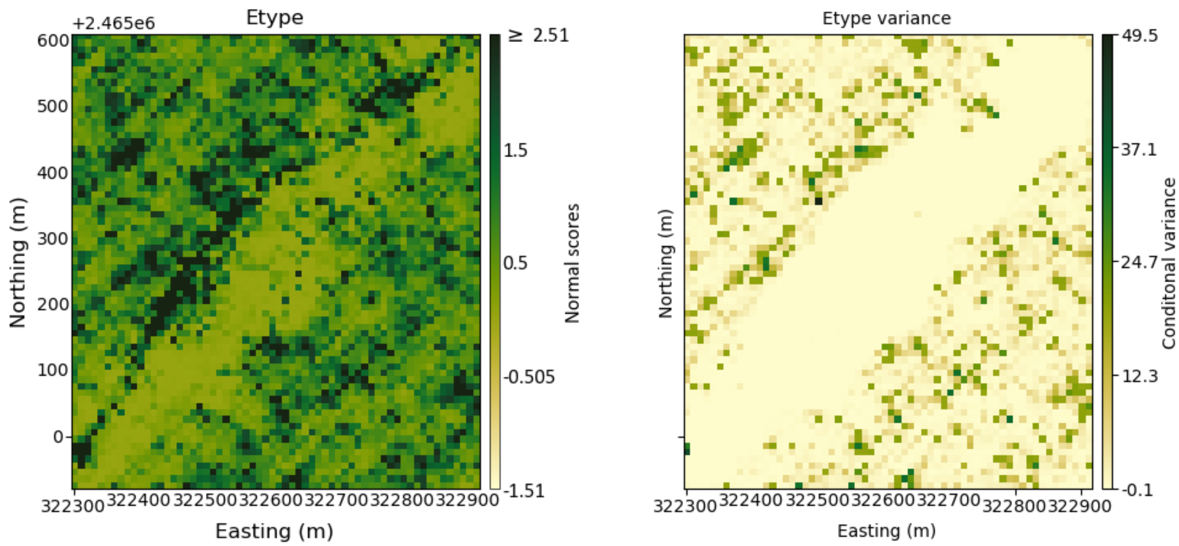
```
Calling:  ['./exes/postsim.exe', 'temp']
```

In [69]:
```
etype_tr = gs.DataFile (outdir + '/simulation/etype_tr.out', griddef = griddef_
etype_tr.head ()
```

Out[69]:

	E-type	variance
0	0.493228	0.292565
1	0.532708	0.358536
2	0.517627	0.204596
3	0.637129	0.755326
4	1.157045	4.281258

In [70]:
```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 2, figsize = (10, 10), sharey = True)
axes = axes.flatten ()

gs.slice_plot (etype_tr, var = 'E-type', orient = 'xy',
               cmap = 'cmo.speed',  title = 'Etype',
               cbar_label = 'Normal scores',
               grid = False, ax = axes [0], unit = 'm',
                  slice_number = 1, vlim = (-1.51, 2.51))
gs.slice_plot (etype_tr, var = 'variance', orient = 'xy',
               cmap = 'cmo.speed',  title = 'Etype variance',
               cbar_label = 'Conditonal variance',
               grid = False, ax = axes [1], unit = 'm',
                  plot_style = True, slice_number = 1)

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 0.95, top = 0.55, wspace = 0.50,
                     hspace = 0.4)
plt.savefig (outdir + 'figures/figure_9.png',
             bbox_inches = 'tight', dpi = 150)
```



Ordinary Kriging Eastimation for AU Samples at Bench 880

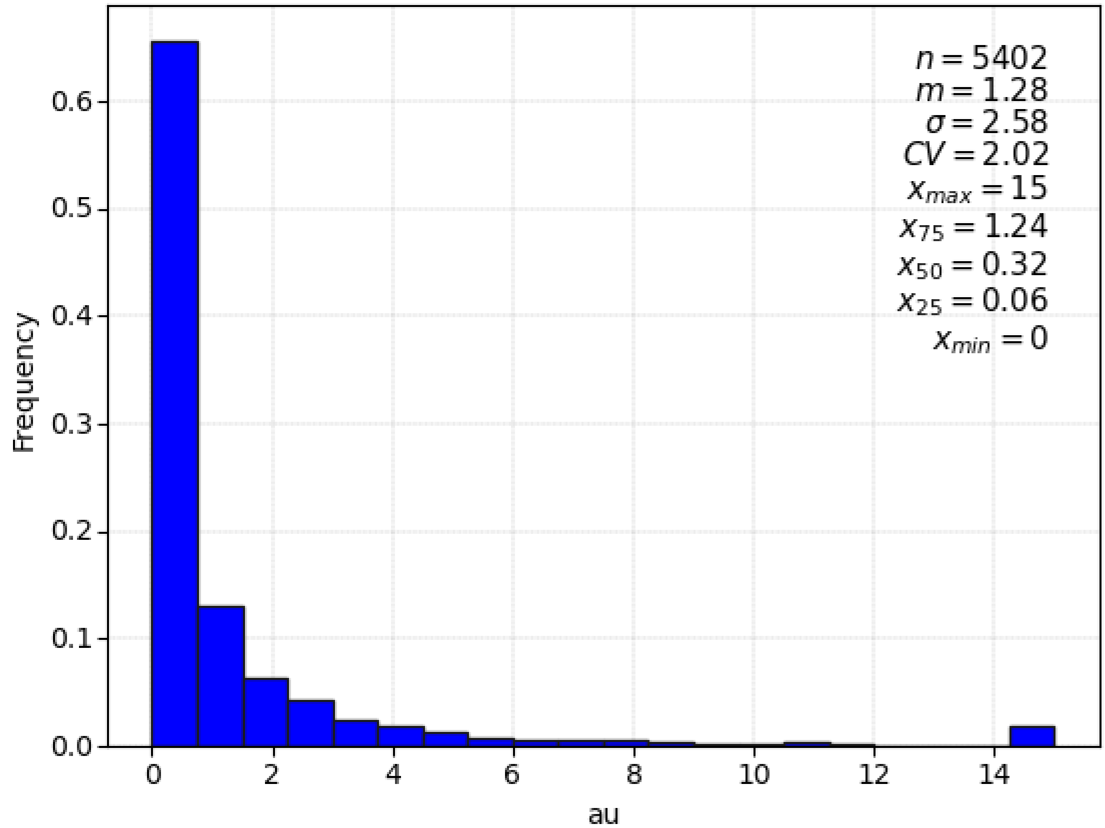
In [71]:
```
b880.head()
```

Out[71]:

	bh	x	y	z	au	Data Spacing (m)
0	1.0	322451.28	2465152.40	877.5	0.084	2.937344
1	2.0	322453.46	2465150.29	877.5	0.103	2.871697
2	3.0	322455.54	2465148.16	877.5	0.044	2.929401
3	4.0	322457.66	2465146.03	877.5	0.096	2.978882
4	5.0	322406.75	2465090.89	877.5	0.321	4.360733

In [72]:
```
gs.histogram_plot (b880, var = 'au',
                          icdf = False, color = 'blue',stat_fontsize = 11, grid = Tr
```

Out[72]: `<Axes: xlabel='au', ylabel='Frequency'>`



Clalculate Experimental Variogram for AU Samples

In [73]:
```
varcalc = gs.Program (program = exedir + 'varcalc', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpwuhmmnbq\varcalc.par has been copied to the clipboard
```

In [74]:
```
varcalcpar = """  Parameters for VARCALC
                  **********************

START OF PARAMETERS:
./datafiles/b_880.out                         - file with data
2   3   4                                           - columns for X, Y, Z coordinat
1   5   0                                           - number of variables,column nu
-998.0    1.0e21                                    - trimming limits
3                                                   - number of directions
45  22.5 9999 0.0 0.0 5.0 0.0                     - Major: azm,azmtol,bandhorz,dip,
30  10  35                                        - number of lags,lag distance,lag
135.0 22.5 999 0.0 0.0 5.0 0.0                     - Minor: azm,azmtol,bandhorz,dip
30  10  35                                         - number of lags,lag distance,la
0.0 0.0 999 -90.0 0.0 5.0 0.0                     - Minor: azm,azmtol,bandhorz,dip,
30 10   5                                        - number of lags,lag distance,lag
./output/experimental/exp1_var_b880_au.out         - file for experimental variogra
0                                                   - legacy output (0=no, 1=write
1                                                   - run checks for common errors
1                                                   - standardize sills? (0=no, 1=y
1                                                   - number of variogram types
1   1   1   ?                                       - tail variable, head variable,




NOTES ON VARIOGRAM CALCULATION:
1) By default, varcalc runs checks for common errors in parameter choices. This
   disabled if desired.
2) Varcalc can standardize using a provided sill (such as a declustered varianc
   For example, if variable 1 has a declustered variance of 8.6, the traditiona
   semivariogram could be standardized by setting the variogram type to:
      1  1  1  8.6
   Alternatively, varcalc can attempt to infer a sill for standardizing by sett
   the variogram type to:
      1  1  1  ?
   The calculated sills will be written to the console.
3) Variogram types are the same as in GSLIB:
      1 = traditional semivariogram
      2 = traditional cross semivariogram
      3 = covariance   (-3 calculates variance (provided sill) -covariance)
      4 = correlogram  (-4 calculates 1-correlation)
      5 = general relative semivariogram
      6 = pairwise relative semivariogram
      7 = semivariogram of logarithms
      8 = semimadogram
      9 = indicator semivariogram - continuous  - requires a cutoff
      10= indicator semivariogram - categorical - requires a category
4) For indicator variograms, the variogram cutoff/categories are specified imme
    1   1   9   1.0   ?               -tail variable, head variable, variogram
5) If desired, the program can write out the variogram points in the gamv2004 s
   for compatibility with older versions. Tilt was not supported in pre-varcalc
   programs so use carefully.
"""

varcalc.run (parstr = varcalcpar, liveoutput = False)
```

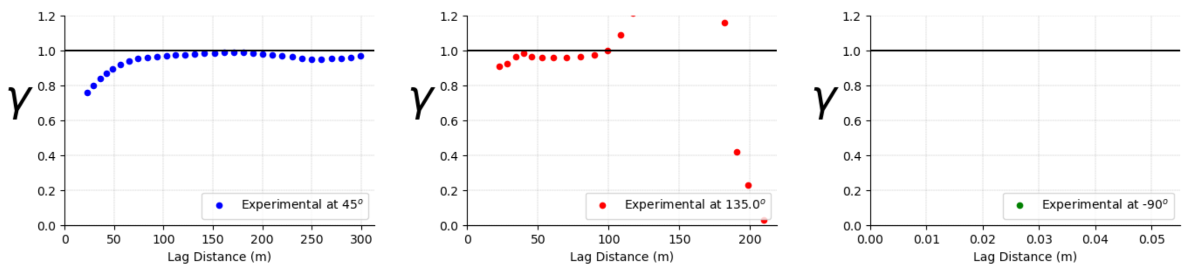
```
Calling:  ['./exes/varcalc', 'temp']
```

In [75]:
```python
exp_var_b880_au = gs.DataFile (outdir + 'experimental/exp1_var_b880_au.out')
exp_var_b880_au.data
```

Out[75]:

	Variogram Index	Lag Distance	Number of Pairs	Variogram Value	Variogram Number	Calculation Azimuth	Calculation Dip	Variogra Ty
0	1.0	23.019419	125321.0	0.757458	1.0	45.0	0.0	
1	1.0	29.470508	201439.0	0.800964	1.0	45.0	0.0	
2	1.0	35.795068	291123.0	0.838860	1.0	45.0	0.0	
3	1.0	42.280881	397593.0	0.868161	1.0	45.0	0.0	
4	1.0	48.869169	513343.0	0.893053	1.0	45.0	0.0	
5	1.0	56.615814	622526.0	0.919637	1.0	45.0	0.0	
6	1.0	65.291018	721421.0	0.938016	1.0	45.0	0.0	
7	1.0	74.406767	813429.0	0.952495	1.0	45.0	0.0	
8	1.0	83.777671	900876.0	0.960526	1.0	45.0	0.0	
9	1.0	93.196168	982967.0	0.964322	1.0	45.0	0.0	
10	1.0	102.856145	1058103.0	0.968305	1.0	45.0	0.0	

In [76]:
```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 3, figsize = (7, 3))
axes = axes.flatten ()

lab = ['Experimental at $45^{o}$', 'Experimental at $135.0^{o}$', 'Experimenta
colors = ['blue', 'red', 'green']

for i, ax, j, m in zip (lab, axes, range (3), colors):
    j = j + 1
    gs.variogram_plot (exp_var_b880_au.data, index = j,
                       grid = True,
                       color = m, minpairs = True, label = i,
                       experimental = True,
                        ax = ax, axis_xy = True, unit = 'm',
                       ms = 5.5, pairnumbers = False)
    ax.legend (loc = 4, fontsize = 10)
    ax.xaxis.label.set_size (10)

fig.tight_layout ()

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 1.95, top = 0.9, wspace = 0.3,
                     hspace = 0.55)

plt.savefig (outdir + 'figures/figure_3.png',
             bbox_inches = 'tight', dpi = 150)
```



In [77]:
```
varmodel = gs.Program (program = exedir + 'varmodel', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpuhoz2er6\varmodel.par has been copied to the clipboard
```

In [78]:
```
varmodelpar = """ Parameters for VARMODEL
                  **********************

START OF PARAMETERS:
{output}                                              - file for modeled variogram poi
3                                                     - number of directions to model 
  45.0 0.0  5500   0.5                                - azm, dip, npoints, point separ
  135.0 0.0  5500   0.5                               - azm, dip, npoints, point separ
  0.0  90.0  180   0.5                                - azm, dip, npoints, point separ
2    0.6                                              - nst, nugget effect
1    0.35  45   0.0   0.0                            - it,cc,ang1,ang2,ang3
       80  40.0  0.01                            - a_hmax, a_hmin, a_vert
1    0.05 45   0.0   0.0                            - it,cc,ang1,ang2,ang3
       1000.0 1000.0 0.03                                 - a_hmax, a_hmin, a_vert
0   100000                                            - fit model (0=no, 1=yes), maxim
1.0                                                   - variogram sill (can be fit, bu
1                                                     - number of experimental files t
varcalc.out                                           - experimental output file 1
2   1   4                                             - # of variograms (<=0 for all),
1   1   10                                            - # pairs weighting, inverse dis
0     10.0                                            - fix Hmax/Vert anis. (0=no, 1=y
0      1.0                                            - fix Hmin/Hmax anis. (0=no, 1=y
varmodelfit.var                                       - file to save fit variogram mod



NOTES ON VARIOGRAM FITTING:
1) This program can be run as the GSLIB program vmodel where an already
   fit variogram model is provided.
2) Alternatively, a variogram model can be fit. Any parameter, except
   the number of structures can be fit. Fitting variogram angles
   is NOT recommended best practice. Options for fitting are:
   ? - fit the parameter with no constraints
   a:b - fit the parameter between a and b
   a: - fit the parameter so it is >=a
   :b - fit the parameter so it is <=b
   There must be no spaces in a:b!
3) Structure types (it) are:
   1 - spherical variogram model
   2 - exponential variogram model
   3 - gaussian variogram model
   4 - hole effect variogram model (cannot be automatically fit)
"""
pars = dict (output = outdir + 'model/varmodel_dirc_b880 _au.out')

varmodel.run (parstr = varmodelpar.format (**pars),
              liveoutput = False)

varmodel_dirc = gs.DataFile (outdir + 'model/varmodel_dirc_b880 _au.out', read
varmodel_dirc.head ()
```

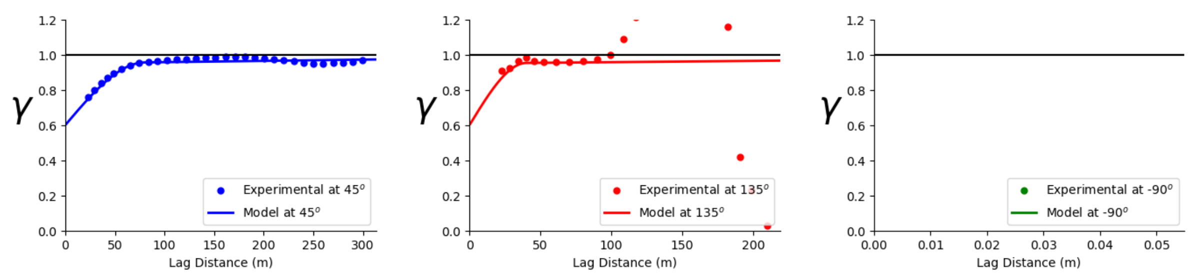
```
Calling:  ['./exes/varmodel', 'temp']
```

Out[78]:

	Variogram Index	Lag Distance	Number of Pairs	Variogram Value	Variogram Number	Calculation Azimuth	Calculation Dip
0	1.0	0.5	1.0	0.603319	1.0	45.0	0.0
1	1.0	1.0	1.0	0.606637	1.0	45.0	0.0
2	1.0	1.5	1.0	0.609955	1.0	45.0	0.0
3	1.0	2.0	1.0	0.613272	1.0	45.0	0.0
4	1.0	2.5	1.0	0.616588	1.0	45.0	0.0

In [79]:

```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 3, figsize = (7, 3))
axes = axes.flatten ()
lab = ['Experimental at $45^{o}$', 'Experimental at $135^{o}$', 'Experimental a
lab_2 = ['Model at $45^{o}$', 'Model at $135^{o}$', 'Model at -$90^{o}$']
colors = ['blue', 'red', 'green']

for i, ax, j, k, d in zip (lab, axes, range (3), lab_2, colors):
    j = j + 1
    gs.variogram_plot (exp_var_b880_au.data, index = j,
                       color = d, minpairs = False,
                       label = i, experimental = True,
                       ax = ax, axis_xy = True, unit = 'm',
                       ms = 5, plot_style = True)

    gs.variogram_plot (varmodel_dirc.data, index = j, sill = False,
                       label = k, experimental = False,
                       ax = ax, axis_xy = True, unit = 'm',
                       ms = 0, lw = 2, color = d,
                       plot_style = True)
    ax.legend (loc = 4, fontsize = 10)
    ax.xaxis.label.set_size (10)

fig.tight_layout ()
plt.subplots_adjust (left = 0.125, bottom = 0.1, right = 1.95, top = 0.9, wspa
plt.savefig (outdir + 'figures/figure_10.png', bbox_inches = 'tight', dpi = 15
```



Save the parameters of the model fitted to the experimental variogram of the Au grades

The model given in the following cell will be used as a structural model required by the kriging program.

In [80]:
```
b880_au_vario ='''2    0.6                                             - nst, nugget e
1    0.35   45    0.0    0.0                       - it,cc,ang1,ang2,ang3
       80   40.0   0.01                        - a_hmax, a_hmin, a_vert
1    0.05 45    0.0    0.0                        - it,cc,ang1,ang2,ang3
       1000.0 1000.0 0.03                             - a_hmax, a_hmin, a_vert'''
print (b880_au_vario)
```

```
2    0.6                                  - nst, nugget effect
1    0.35   45    0.0    0.0             - it,cc,ang1,ang2,ang3
       80   40.0   0.01              - a_hmax, a_hmin, a_vert
1    0.05 45    0.0    0.0              - it,cc,ang1,ang2,ang3
       1000.0 1000.0 0.03                   - a_hmax, a_hmin, a_vert
```

OK Estimation

Load the `kt3dn.exe` `GSLIB` program and its parameter file

In [82]:
```
blksize = (10, 10, 5)
griddef_b880 = b880.infergriddef (blksize)
griddef_b880
```

Out[82]:
```
Pygeostat GridDef:
62 322301.0 10.0
69 2464926.0 10.0
3 874.5 5.0
```

In [83]:
```
kt3dn = gs.Program (program = exedir + 'kt3dn.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmpkuqbfkta\kt3dn.par has been copied to the clipboard
```

In [84]:
```
start_workflow_time = time.time()

kt3dnpar = """   Parameters for KT3DN
                  ********************
START OF PARAMETERS:
{datafl}                         - file with data
1  2  3  4  5  0                 - columns for DH,X,Y,Z,var,sec var
-998.0    1.0e21                 - trimming limits
0                                - option: 0=grid, 1=cross, 2=jackknife
nofile                           - file with jackknife data
1   2   0    3    0              - columns for X,Y,Z,vr and sec var
nofile                           - data spacing analysis output file (see note
0    15.0                        - number to search (0 for no dataspacing analy
0    100   0                     - debugging level: 0,3,5,10; max data for GSK\
nofile                           - file for debugging output (see note)
{output}                         - file for kriged output (see GSB note)
{grid}
3    3      3                    - x,y and z block discretization
0     25     12    1             - min, max data for kriging,upper max for ASO
0      0                         - max per octant, max per drillhole (0-> not u
80.0  40.0  5.0                  - maximum search radii
 45.0   0.0   0.0                 - angles for search ellipsoid
1                                 - 0=SK,1=OK,2=LVM(resid),3=LVM((1-w)*m(u))),4
0.0   0.0  0.0                    - mean (if 0,4,5,6), corr. (if 4 or 6), var.
0 0 0 0 0 0 0 0 0                - drift: x,y,z,xx,yy,zz,xy,xz,zy
0                                - 0, variable; 1, estimate trend
nofile                           - gridded file with drift/mean
4                                - column number in gridded file
nofile                           - gridded file with keyout (see note)
0    1                           - column (0 if no keyout) and value to keep
{varmodel}

Data spacing analysis explained:
The approximate data spacing (given a number of neighbours to use
and composite length for 3D kriging) is calculated as well as the distance to t
neighbour. Results are summarized in the data spacing output file.

GSB explained:
Input and output grid files may be binary when running the program in grid mode
This includes the optonal external and keyout files, as well as the output file
assumes that a file is formatted as GSLIB-style binary (GSB) if a .gsb extensio

Debugging levels explained:
0: no debugging output
3: get extra kriging information
5: get the above + GSK variance
10: get the above + kriging matrix info
Note1: idbg>=5 may involve a long run time
Note2: GSKV calculation uses search radius

Auto Search Optimization (ASO) explained:
Set min data for kriging to the negative of value
         ie: for min data = 2, set min data to -2
max data is then the lower limit for number of data
Provide a upper limit and increment
         ie: -4  8  12  1 means that
         4 is the min data for kriging
```

```
         and kriging will be performed with a max of
         8, 9, 10, 11 and 12 data sequentially
The outputs are output as realizations from
the lower limit to the upper limit (similar to SGSIM)

Keyout explained:
A keyout variable, matching the dimensions of the output grid, may be used to s
whether each location should be estimated (value to keep) or unestimated (every
This has the potential to substatially increase execution speed when only inte
irregular sub-volume of the modeling grid.

Invdist option explained:
Set nst to -1 for inverse distance
Inverse distance estimates are calculated with the anisotropy/angle informatio
 weight(i)*=1/(h+c0(1))^(cc(1))
The ID weights are then scaled to sum to 1
"""
kt3dn.run (parstr = kt3dnpar.format (datafl = datadir + 'b_880.out',
                                     grid = griddef_b880,
                                    output = outdir + '/estimation/kt3dn_b880.
                                    varmodel = b880_au_vario), liveoutput = Fa
```

```
Calling:  ['./exes/kt3dn.exe', 'temp']
```

In [85]:

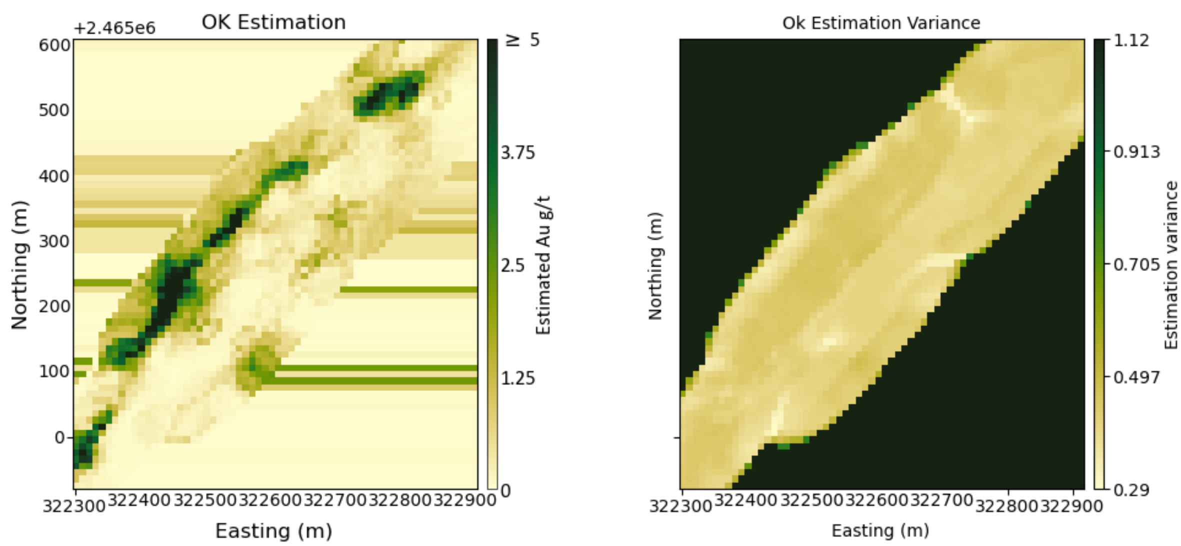
```
Ok_estim = gs.DataFile (outdir + '/estimation/kt3dn_b880.out',
                        griddef = griddef_b880, readfl = True)
Ok_estim.head (n = 5)
```

Out[85]:

	Estimate	EstimationVariance
0	0.210695	0.438096
1	0.162935	0.434080
2	0.184026	0.433308
3	0.096740	0.419047
4	0.088697	0.398959

Display Slices

In [86]:
```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 2, figsize = (10, 10), sharey = True)
axes = axes.flatten ()

gs.slice_plot (Ok_estim, var = 'Estimate', orient = 'xy',
               cmap = 'cmo.speed',  title = 'OK Estimation',
               cbar_label = 'Estimated Au g/t',
               grid = False, ax = axes [0], unit = 'm',
                  slice_number = 1, vlim = (0, 5))
gs.slice_plot (Ok_estim, var = 'EstimationVariance', orient = 'xy',
               cmap = 'cmo.speed',  title = 'Ok Estimation Variance',
               cbar_label = 'Estimation variance',
               grid = False, ax = axes [1], unit = 'm',
                  plot_style = True, slice_number = 1)

plt.subplots_adjust (left = 0.125, bottom = 0.1,
                     right = 0.95, top = 0.55, wspace = 0.50,
                     hspace = 0.4)
plt.savefig (outdir + 'figures/figure_9.png',
             bbox_inches = 'tight', dpi = 150)
```



Cross Validation

In [87]:
```
kt3dn = gs.Program (program = exedir + 'kt3dn.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmp8bur30rd\kt3dn.par has been copied to the clipboard
```

In [88]:
```
kt3dnpar = """   Parameters for KT3DN
                   ********************
START OF PARAMETERS:
{datafl}                          - file with data
1  2  3  4  5  0                  - columns for DH,X,Y,Z,var,sec var
-998.0    1.0e21                  - trimming limits
1                                 - option: 0=grid, 1=cross, 2=jackknife
nofile                            - file with jackknife data
1   2   0    3    0               - columns for X,Y,Z,vr and sec var
nofile                            - data spacing analysis output file (see note
0    15.0                         - number to search (0 for no dataspacing analy
0    100   0                      - debugging level: 0,3,5,10; max data for GSKV
nofile                            - file for debugging output (see note)
{output}                          - file for kriged output (see GSB note)
{grid}
3    3      3                     - x,y and z block discretization
0    25    12    1                - min, max data for kriging,upper max for ASO,
0      0                          - max per octant, max per drillhole (0-> not u
1000.0  1000.0  30.0              - maximum search radii
 0.0   0.0   0.0                  - angles for search ellipsoid
1                                 - 0=SK,1=OK,2=LVM(resid),3=LVM((1-w)*m(u))),4=
0.0  0.0  0.0                     - mean (if 0,4,5,6), corr. (if 4 or 6), var. r
0 0 0 0 0 0 0 0 0                 - drift: x,y,z,xx,yy,zz,xy,xz,zy
0                                 - 0, variable; 1, estimate trend
nofile                            - gridded file with drift/mean
4                                 - column number in gridded file
nofile                            - gridded file with keyout (see note)
0    1                            - column (0 if no keyout) and value to keep
{varmodel}

Data spacing analysis explained:
The approximate data spacing (given a number of neighbours to use
and composite length for 3D kriging) is calculated as well as the distance to t
neighbour. Results are summarized in the data spacing output file.

GSB explained:
Input and output grid files may be binary when running the program in grid mode
This includes the optonal external and keyout files, as well as the output file
assumes that a file is formatted as GSLIB-style binary (GSB) if a .gsb extensio

Debugging levels explained:
0: no debugging output
3: get extra kriging information
5: get the above + GSK variance
10: get the above + kriging matrix info
Note1: idbg>=5 may involve a long run time
Note2: GSKV calculation uses search radius

Auto Search Optimization (ASO) explained:
Set min data for kriging to the negative of value
         ie: for min data = 2, set min data to -2
max data is then the lower limit for number of data
Provide a upper limit and increment
         ie: -4  8  12  1 means that
         4 is the min data for kriging
         and kriging will be performed with a max of
         8, 9, 10, 11 and 12 data sequentially
```

```
The outputs are output as realizations from
the lower limit to the upper limit (similar to SGSIM)

Keyout explained:
A keyout variable, matching the dimensions of the output grid, may be used to s
whether each location should be estimated (value to keep) or unestimated (every
This has the potential to substatially increase execution speed when only inter
irregular sub-volume of the modeling grid.

Invdist option explained:
Set nst to -1 for inverse distance
Inverse distance estimates are calculated with the anisotropy/angle informatior
 weight(i)*=1/(h+c0(1))^(cc(1))
The ID weights are then scaled to sum to 1
"""
kt3dn.run (parstr = kt3dnpar.format (datafl = datadir + 'b_880.out',
                                     grid = griddef_b880,
                                    output = outdir + '/cross_val/kt3dn_ok_b880
                                    varmodel = b880_au_vario), liveoutput = Fal
```

```
Calling:  ['./exes/kt3dn.exe', 'temp']
```

In [89]:

```
Ok_cross = gs.DataFile (outdir + '/cross_val/kt3dn_ok_b880.out',
                        griddef = griddef_b880, readfl = True)
Ok_cross.head (n = 5)
```

Out[89]:

	X	Y	Z	True	Estimate	EstimationVariance	Error: est-true
0	322451.28	2465152.4	877.5	0.084	0.643621	0.229101	0.559621
1	322453.46	2465150.3	877.5	0.103	0.809523	0.227525	0.706523
2	322455.54	2465148.2	877.5	0.044	0.580669	0.224601	0.536669
3	322457.66	2465146.0	877.5	0.096	0.567866	0.221861	0.471866
4	322406.75	2465090.9	877.5	0.321	0.356244	0.212638	0.035244

In [90]:

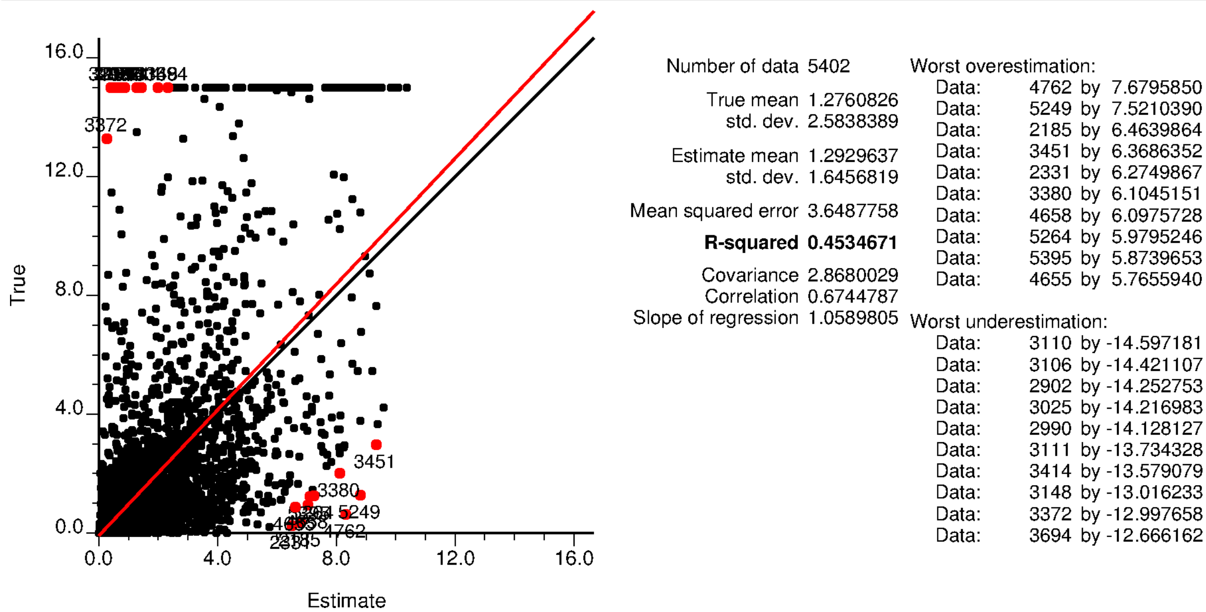
```
scatxval = gs.Program (program = exedir + 'scatxval.exe', getpar = True)
```

```
C:\Users\hugh\Documents\applied statistics\CPAG23_Project_NadirElnour\SIMULAT
ION BLASTHOLE\tmps4pkwos0\scatxval.par has been copied to the clipboard
```

In [91]:
```
scatxvalpar = """ Parameters for SCATXVAL
                  ***********************

START OF PARAMETERS:
{datafl}                          - file with cross validation results
4   5   0                         - columns for true, estimate, ID
-1.0e21   1.0e21                  - trimming limits
{output}                          - file for Postscript output
0                                 - 0=continuous, 1=categorical
0.0 16.66    0                    - minimum, maximum, (0=arith, 1=log)
10                                - number of worst estimates to show
No cut off                        - cutoff value (outside range for missing)
"""

scatxval.run ( parstr = scatxvalpar.format (datafl = outdir + '/cross_val/kt3d
                                    output = outdir + '/cross_val/c
```

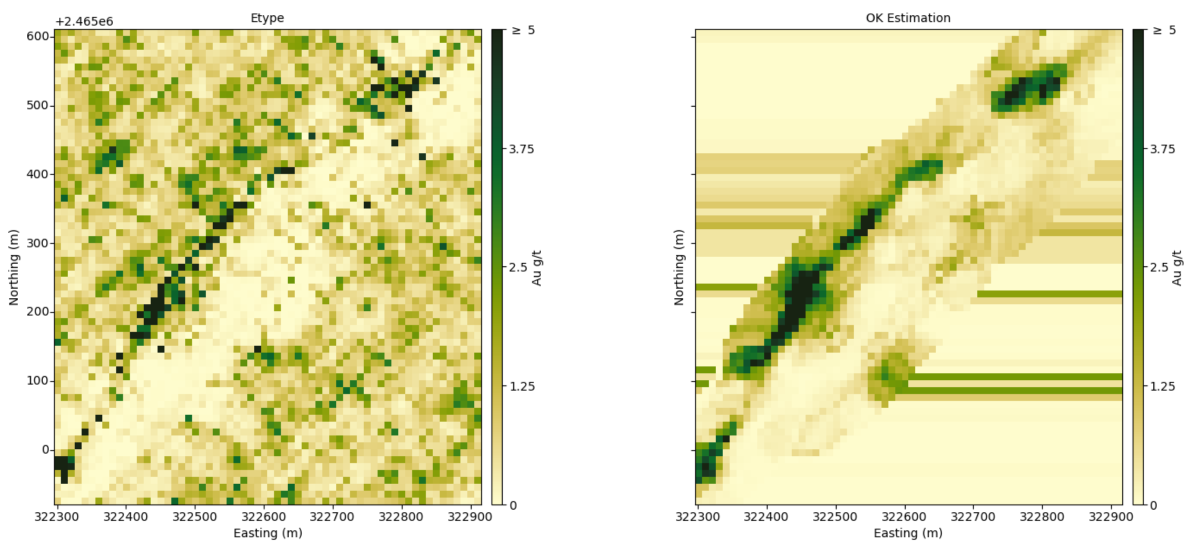
```
Calling:  ['./exes/scatxval.exe', 'temp']
```

In [92]:
```
DisplayPostscript (outdir + '/cross_val/cross_b880.ps')
```



Display the etype model and OK estimate

In [93]:
```
mpl.style.use ('default')
fig, axes = plt.subplots (1, 2, figsize = (15, 15), sharey = True)

axes = axes.flatten ()

gs.slice_plot (etype_tr, var = 'E-type', orient = 'xy', cmap = 'cmo.speed',  t
               cbar_label = 'Au g/t', grid = False, ax = axes [0], unit = 'm',
                  slice_number = 1,vlim = (0, 5), plot_style = True)
gs.slice_plot (Ok_estim, var = 'Estimate', orient = 'xy', cmap = 'cmo.speed',
               cbar_label = 'Au g/t', grid = False, ax = axes [1], unit = 'm',
                   slice_number = 1, vlim = (0, 5), plot_style = True)

plt.subplots_adjust (left = 0.125, bottom = 0.1, right = 0.95, top = 0.55, wsp

plt.savefig (outdir + 'figures/figure_12.png', bbox_inches = 'tight', dpi = 15
```



In [ ]: