# Thesis

October 24, 2024

# 1 Master Thesis Notebook

## 1.1 Setup

[1]:
```
# Importation of the classes and methods associated
from classes import *
```

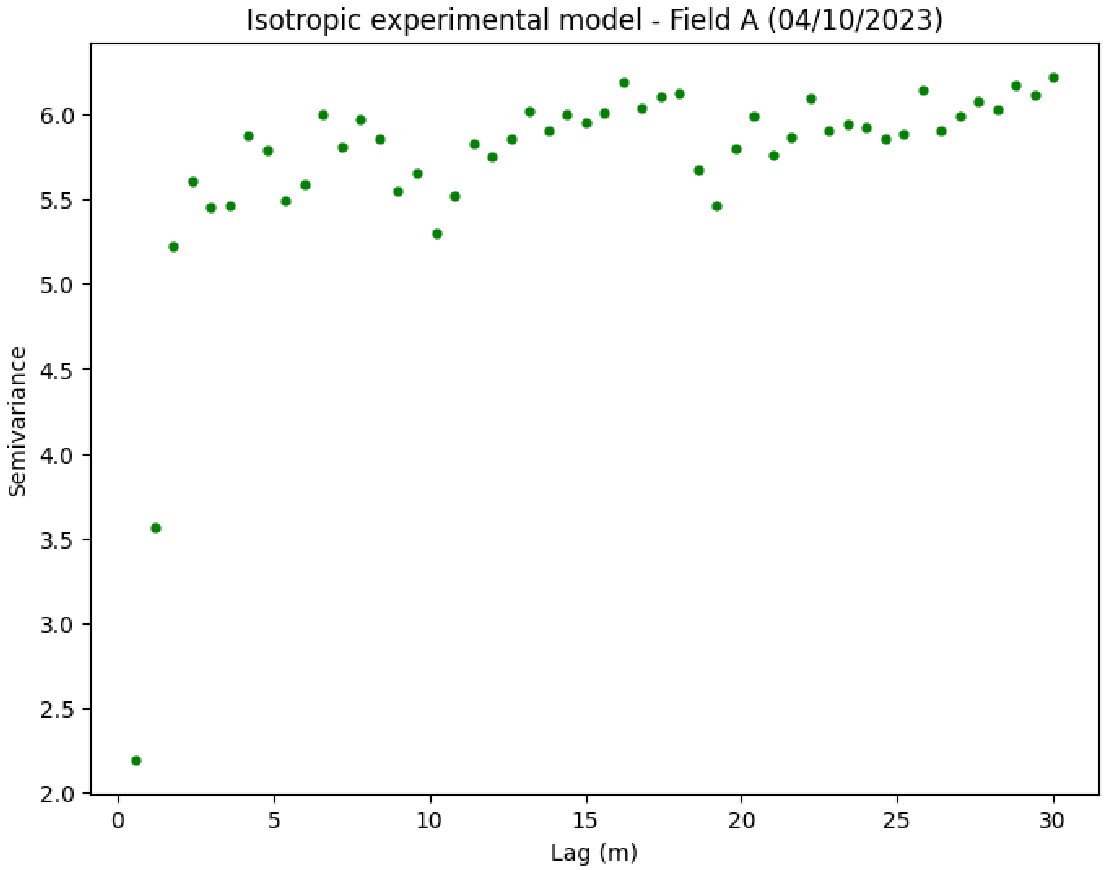
## 1.2 GPR VWC Analysis

### 1.2.1 Choosing the variogram model (field A)

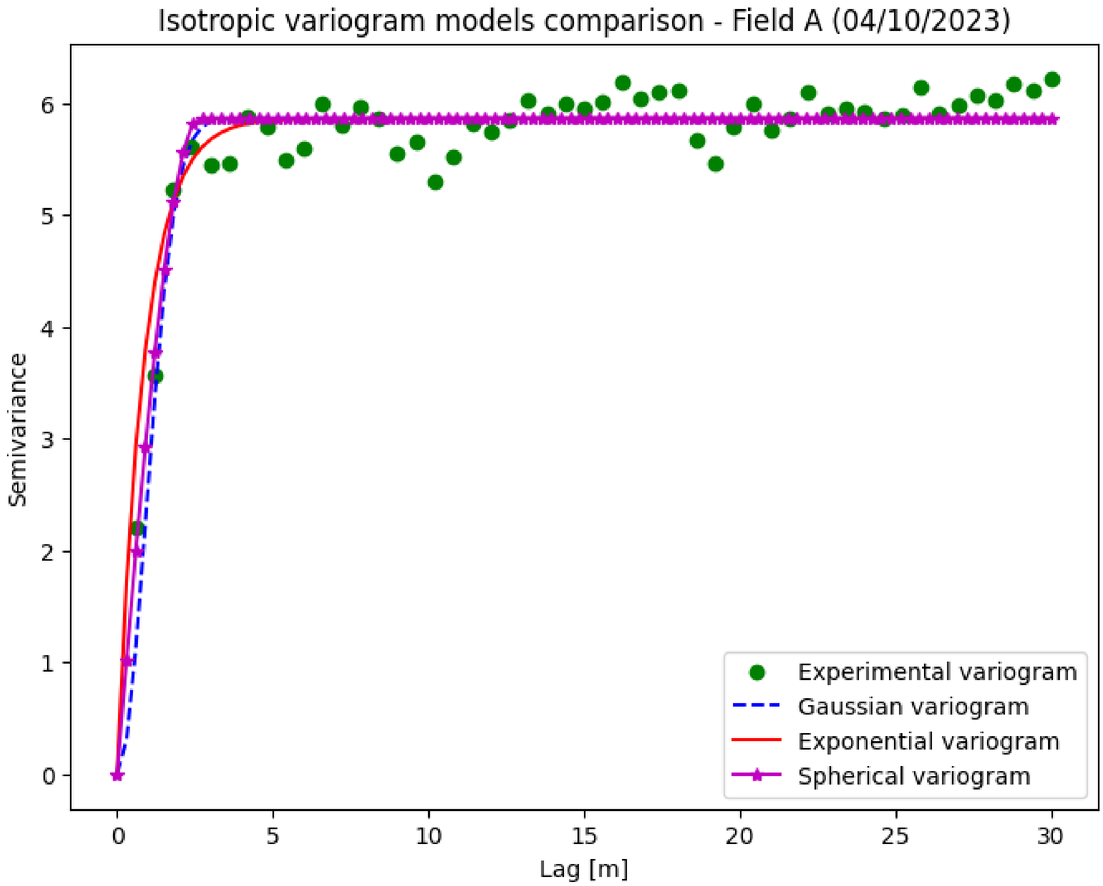
A variogram, also known as a semivariogram, is a measure of spatial variability or spatial dependence of a variable across different locations in a spatial domain. It quantifies how the variance of a variable changes with distance and direction. The variogram is crucial in kriging because it defines the spatial structure or dependence of the variable being estimated.

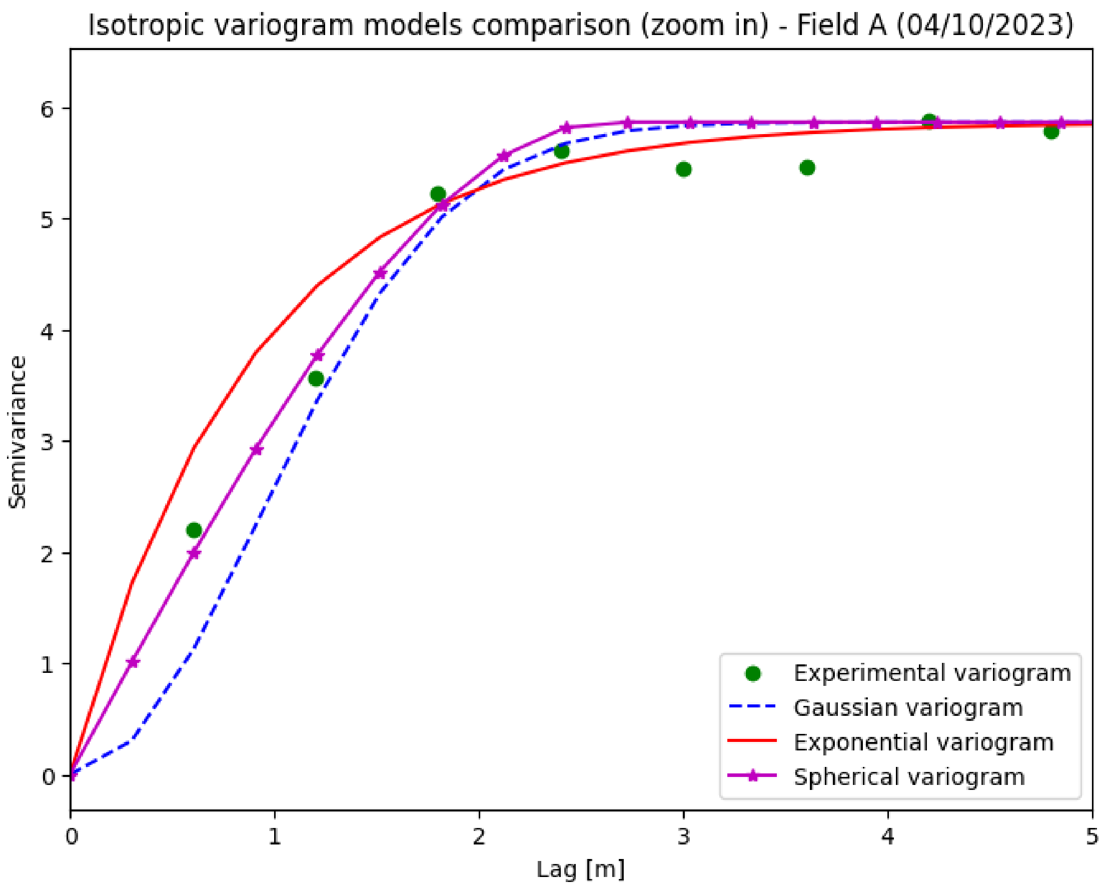
[2]:
```
experimental_vario_a = Variogram(resolution=1, field_letter="A",␣
↪sample_number=6)

experimental_vario_a.determ_experimental_vario() ;
# ; hide output of the cell
```




[3]:
```
experimental_vario_a.fit_models()
```

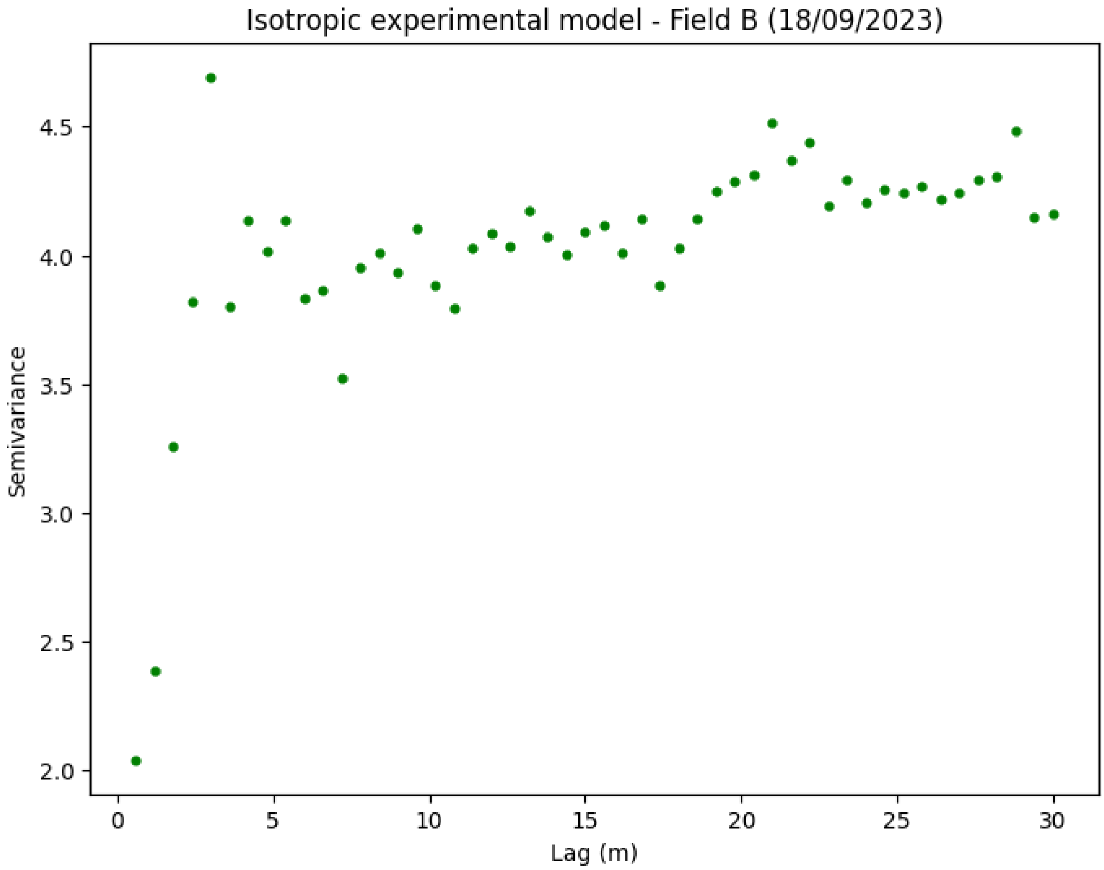

Isotropic variogram models comparison - Field A (04/10/2023)
Semivariance
Lag [m]
Experimental variogram
Gaussian variogram
Exponential variogram
Spherical variogram
0
1
2
3
4
5
6
0
5
10
15
20
25
30



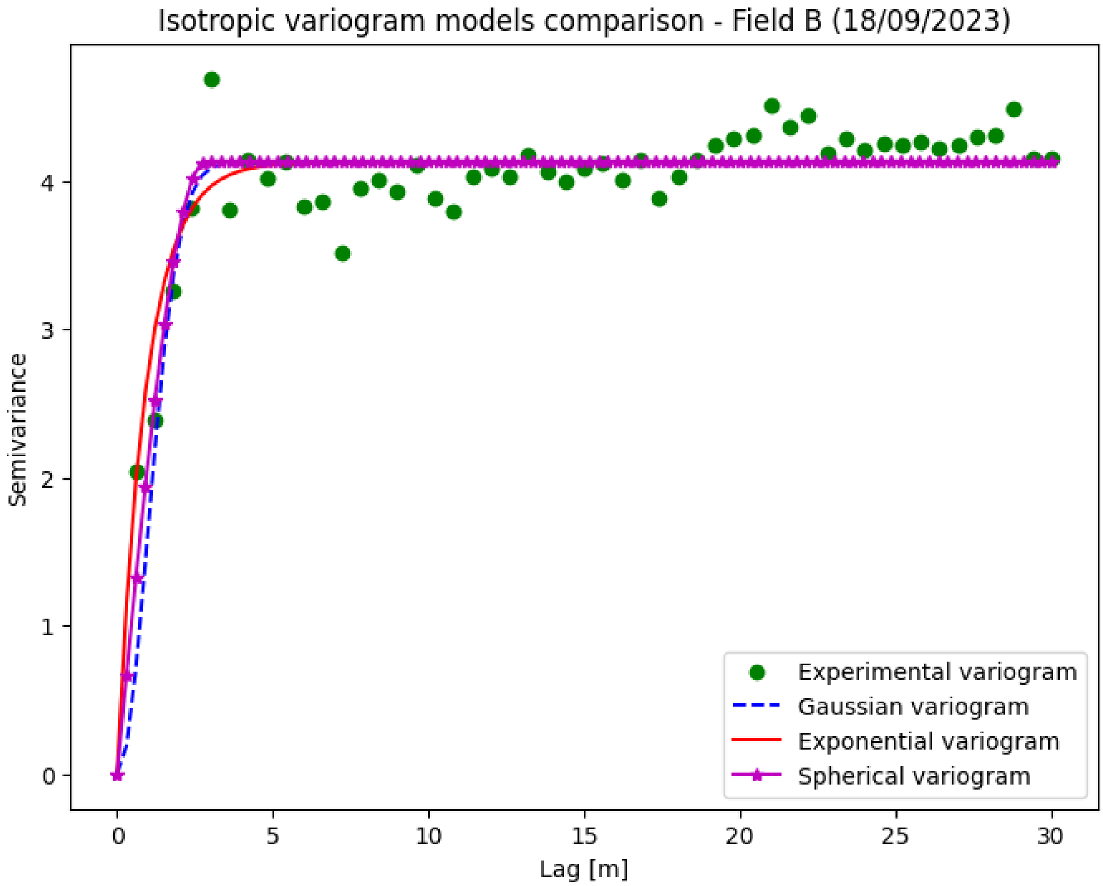

### 1.2.2 Choosing the variogram model (field B)

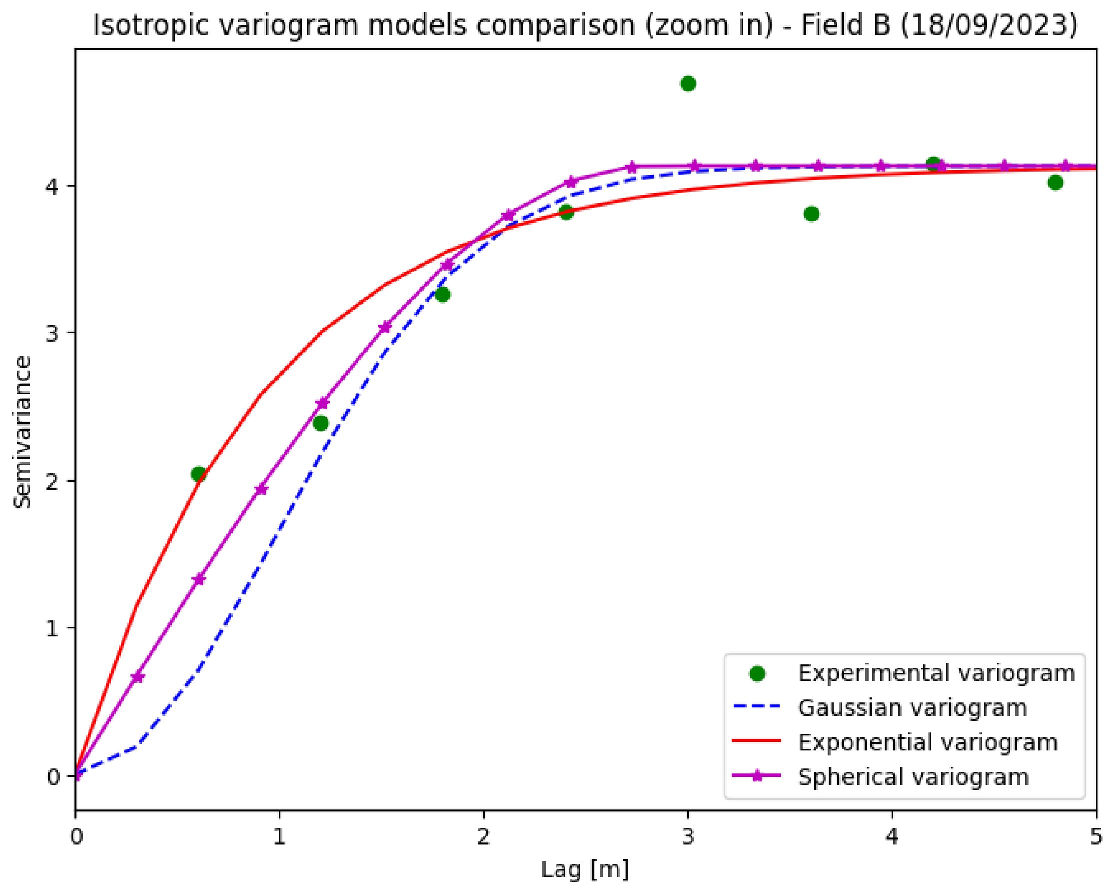
[4]:
```
experimental_vario_b = Variogram(resolution=1, field_letter="B",␣
↪sample_number=5)

experimental_vario_b.determ_experimental_vario() ;
```




[5]:
```
experimental_vario_b.fit_models()
```

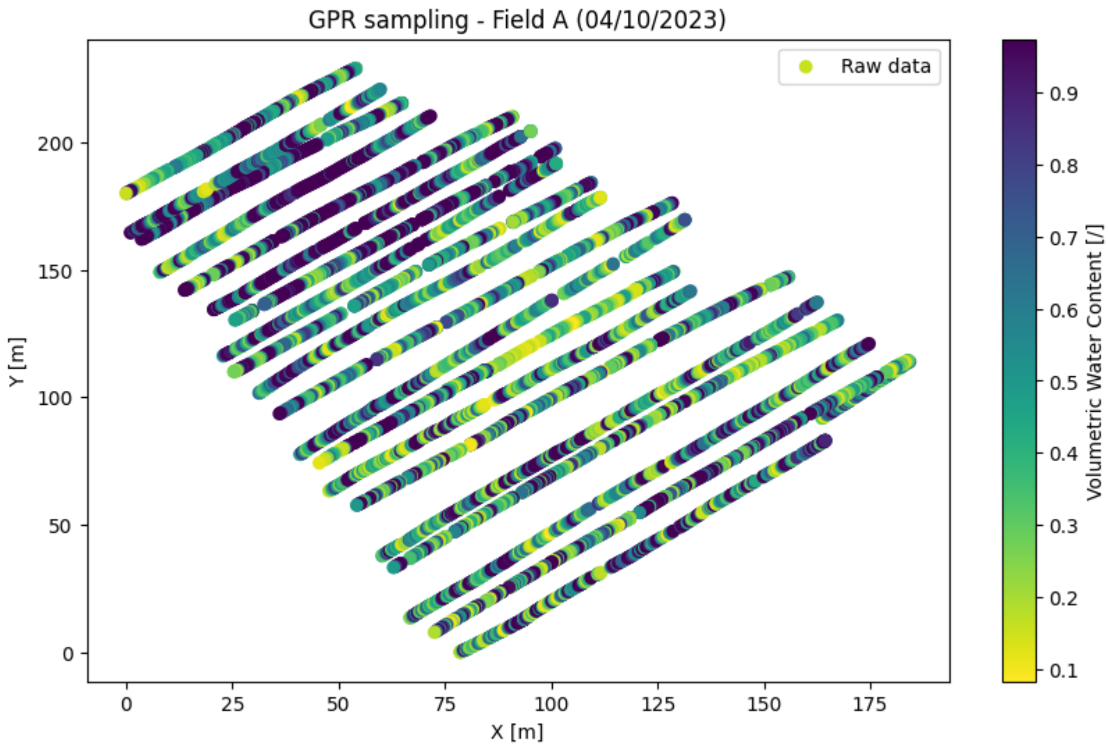

Isotropic variogram models comparison - Field B (18/09/2023)
Semivariance
Lag [m]
0
1
2
3
4
0
5
10
15
20
25
30
Experimental variogram
Gaussian variogram
Exponential variogram
Spherical variogram



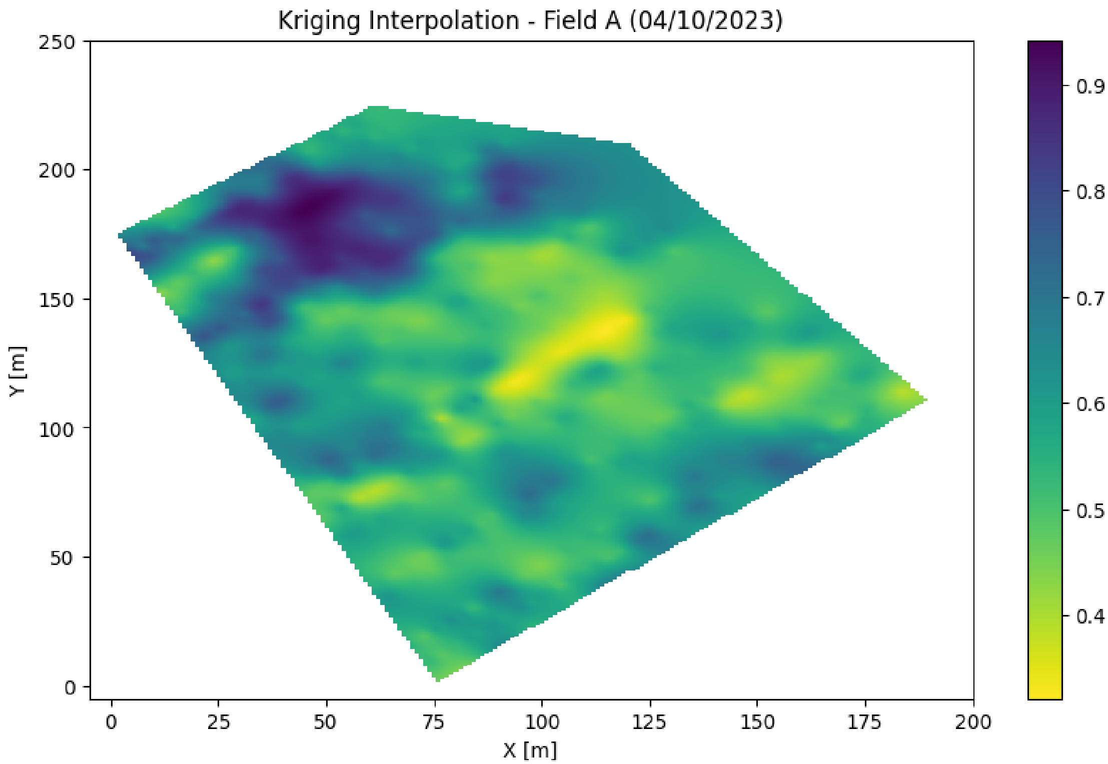

### 1.2.3 Raw data to Kriged data

[2]:
```
# Instance creation
gpr_a = GprAnalysis(field_letter="A", sample_number=6)  # Sample number [0-11]␣
↪# A/B
# Instance creation
gpr_b = GprAnalysis(field_letter="B", sample_number=5)  # Sample number [0-11]␣
↪# A/B
```

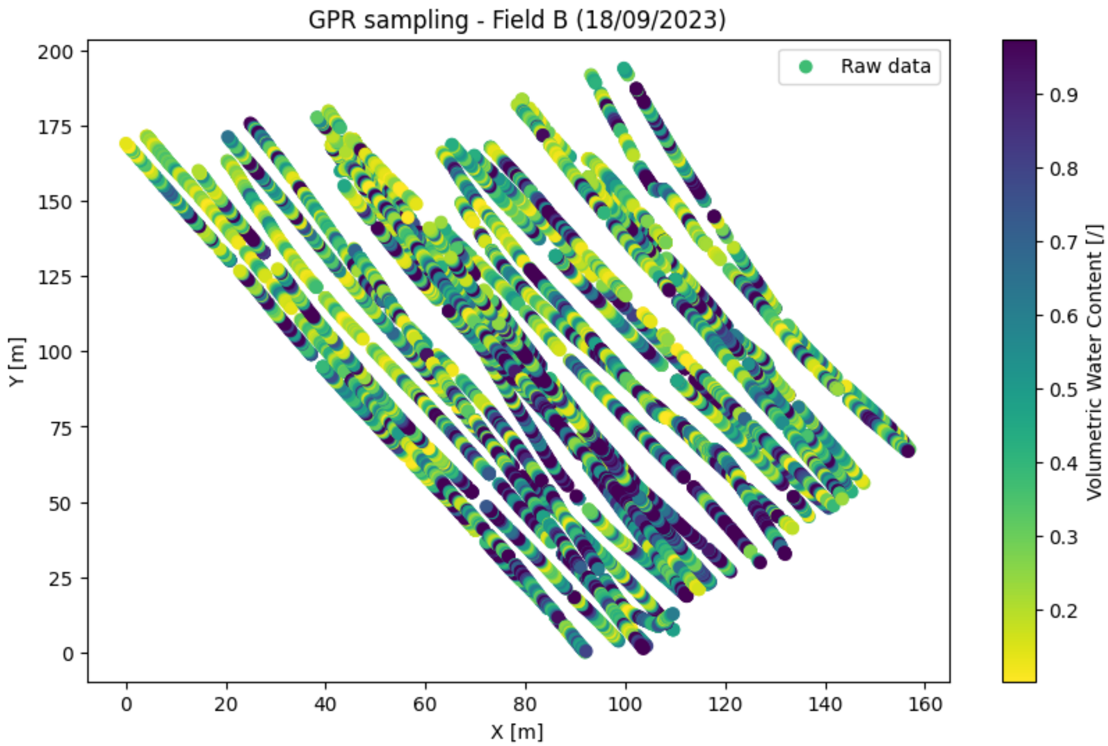
[7]:
```
# Raw Sample
gpr_a.plot_raw_data()
```



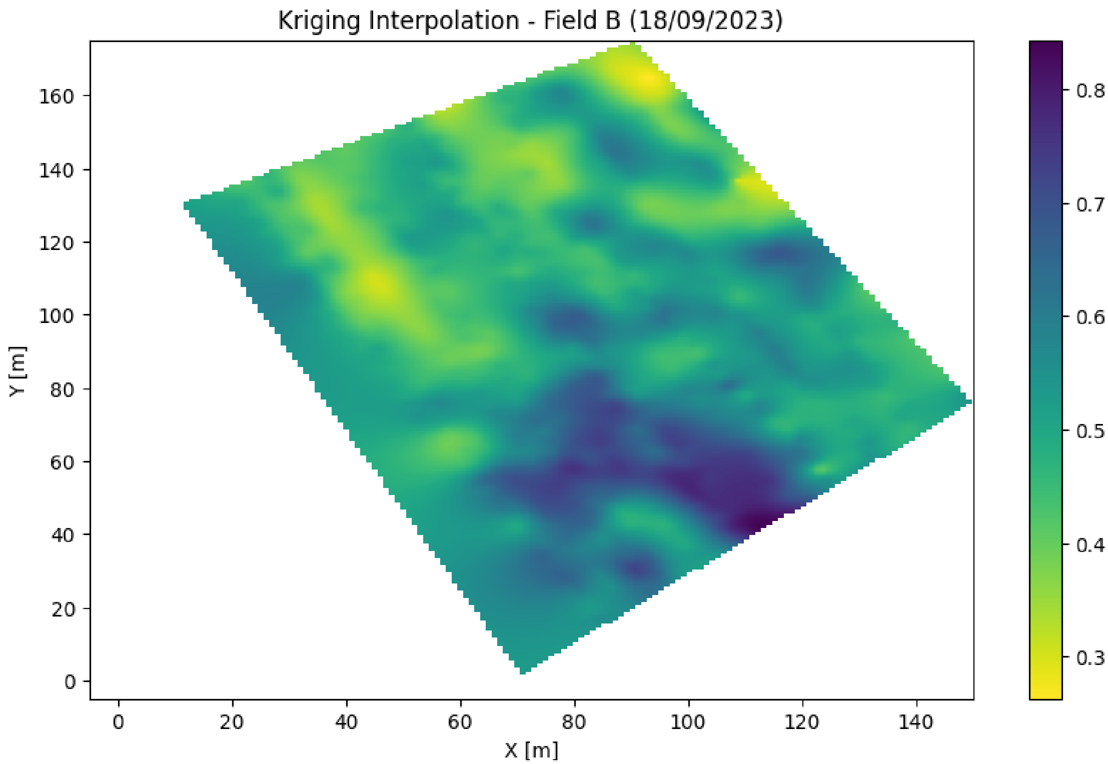

[8]:
```
gpr_a.kriging(plot=True)
```




[3]:
```
gpr_b.plot_raw_data()
```

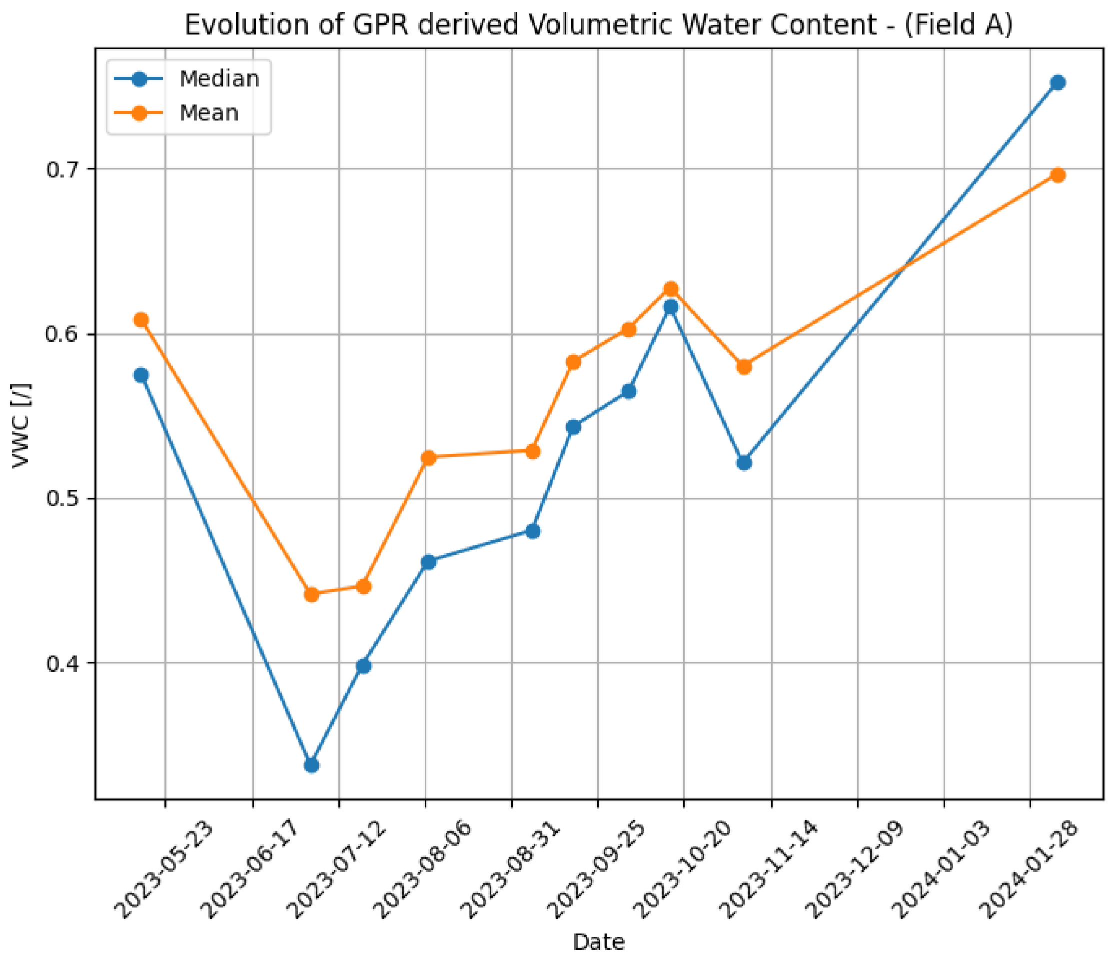



[4]:
```
gpr_b.kriging(plot=True)
```

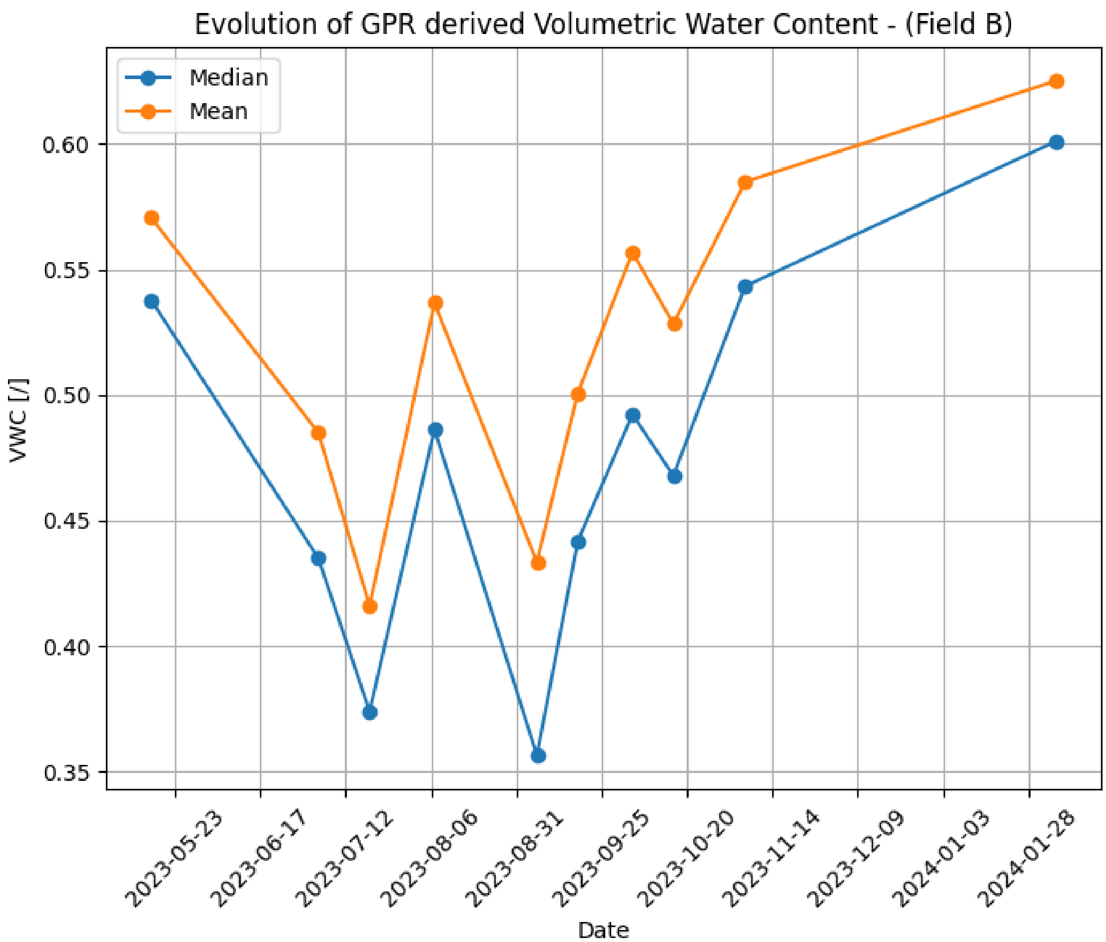



### 1.2.4 GPR derived VWC evolution

[4]:
```
# Mean and median evolution (raw data)
gpr_a.plot_mean_median(plot=True)
```



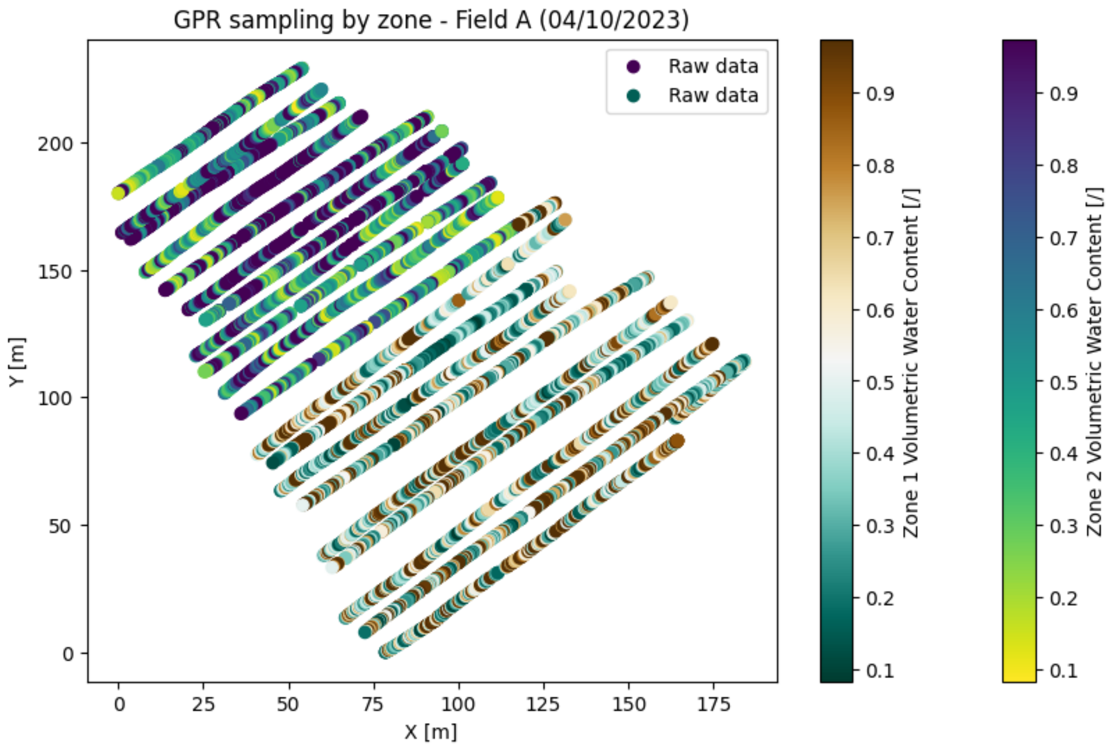

[5]:
```
# Median evolution (raw data)
gpr_b.plot_mean_median(plot=True)
```

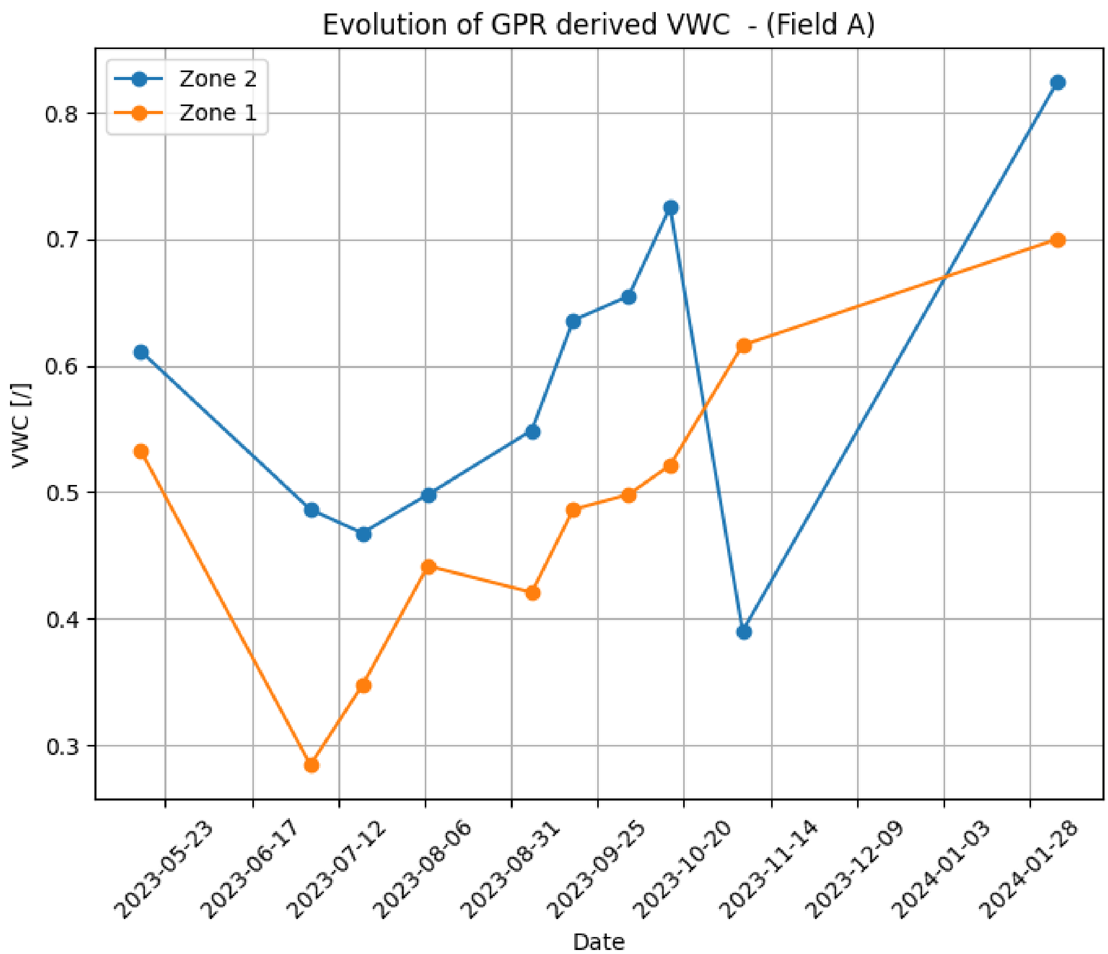



### 1.2.5 Zonal tendencies (field A)

[6]:

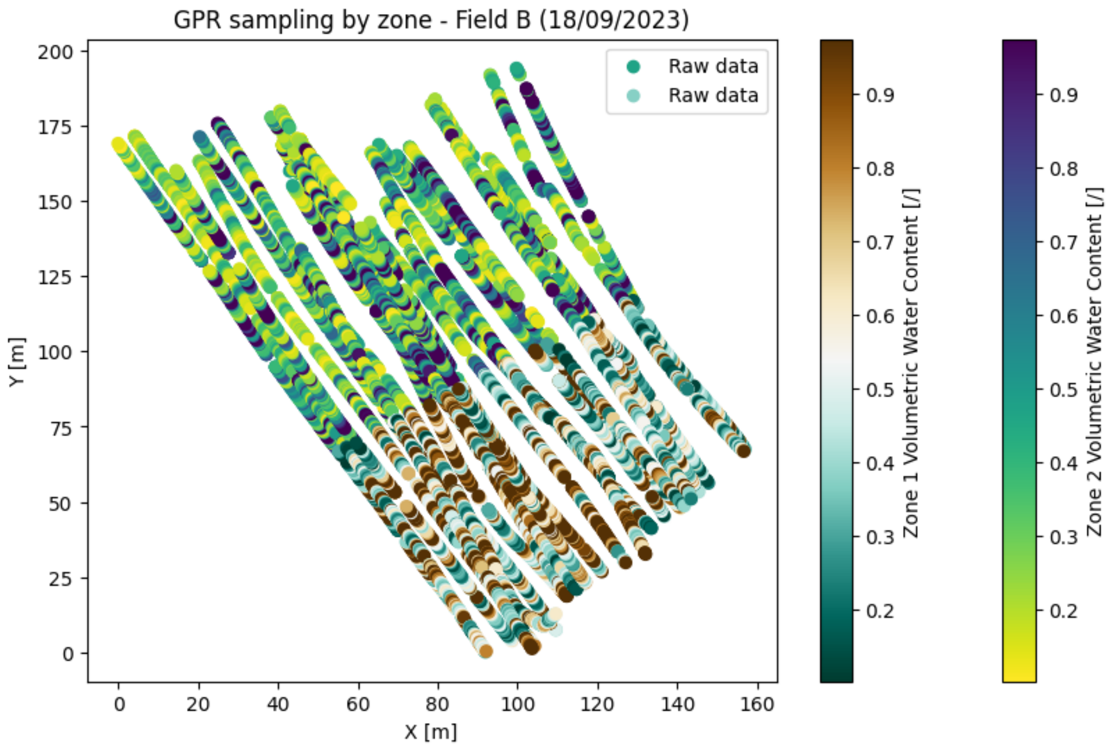
```
gpr_a.zonal_check()
```


GPR sampling by zone - Field A (04/10/2023)
Raw data
Raw data
Y [m]
X [m]
0
50
100
150
200
0
25
50
75
100
125
150
175
Zone 1 Volumetric Water Content [/]
Zone 2 Volumetric Water Content [/]
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

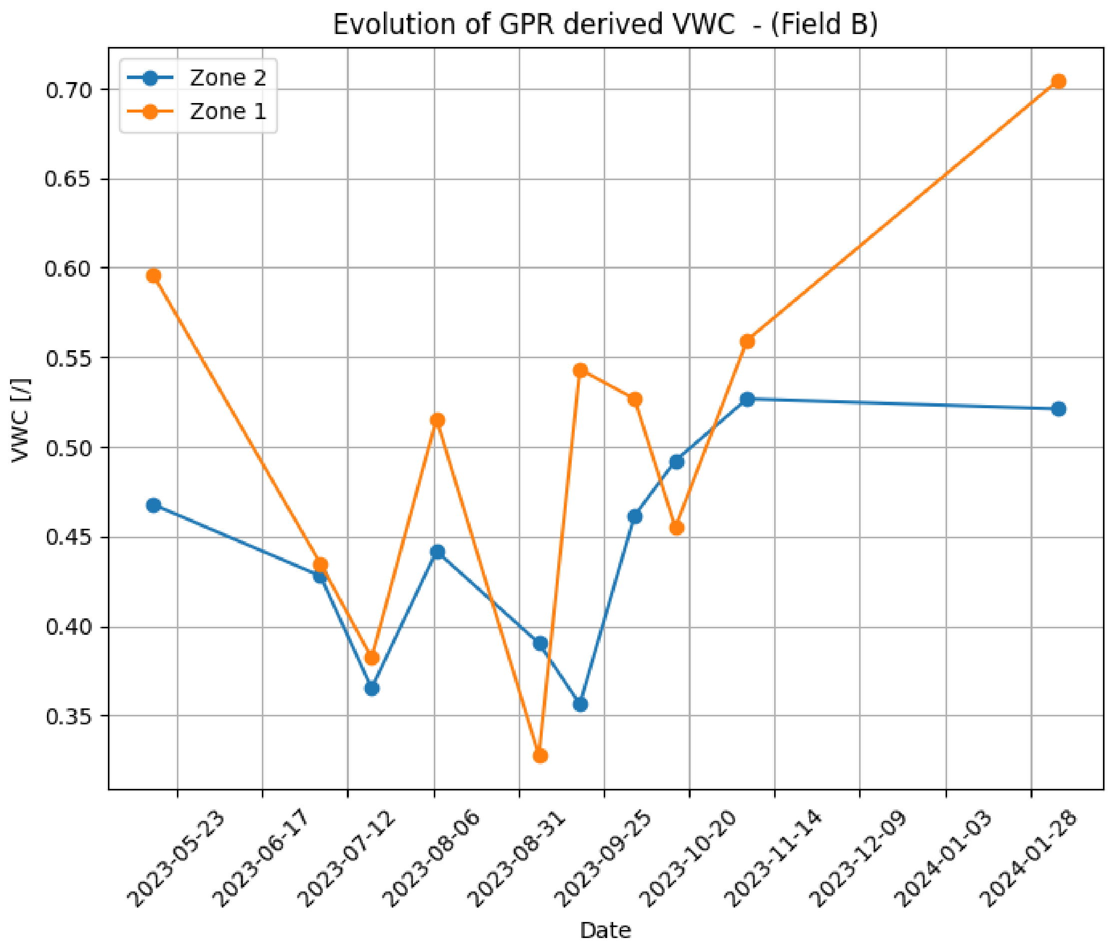



#### 1.2.6 Zonal tendencies (field B)

[7]:
```
gpr_b.zonal_check()
```



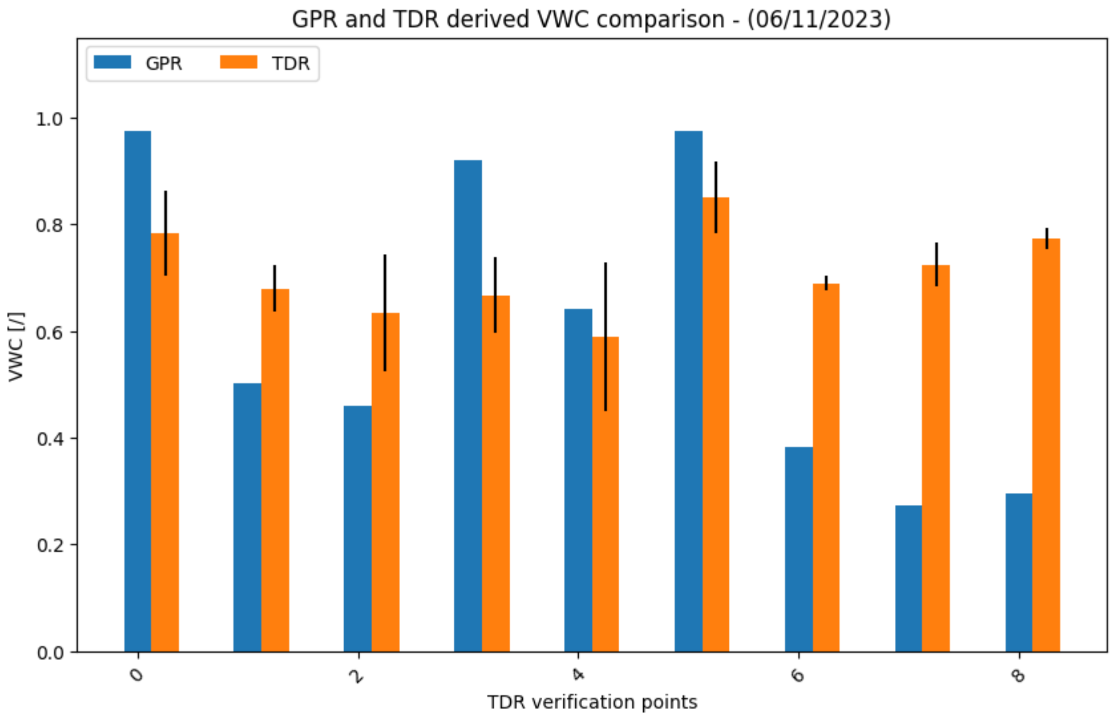
GPR sampling by zone - Field B (18/09/2023)
Raw data
Raw data
Y [m]
X [m]
0
25
50
75
100
125
150
175
200
0
20
40
60
80
100
120
140
160
Zone 1 Volumetric Water Content [/]
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Zone 2 Volumetric Water Content [/]
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

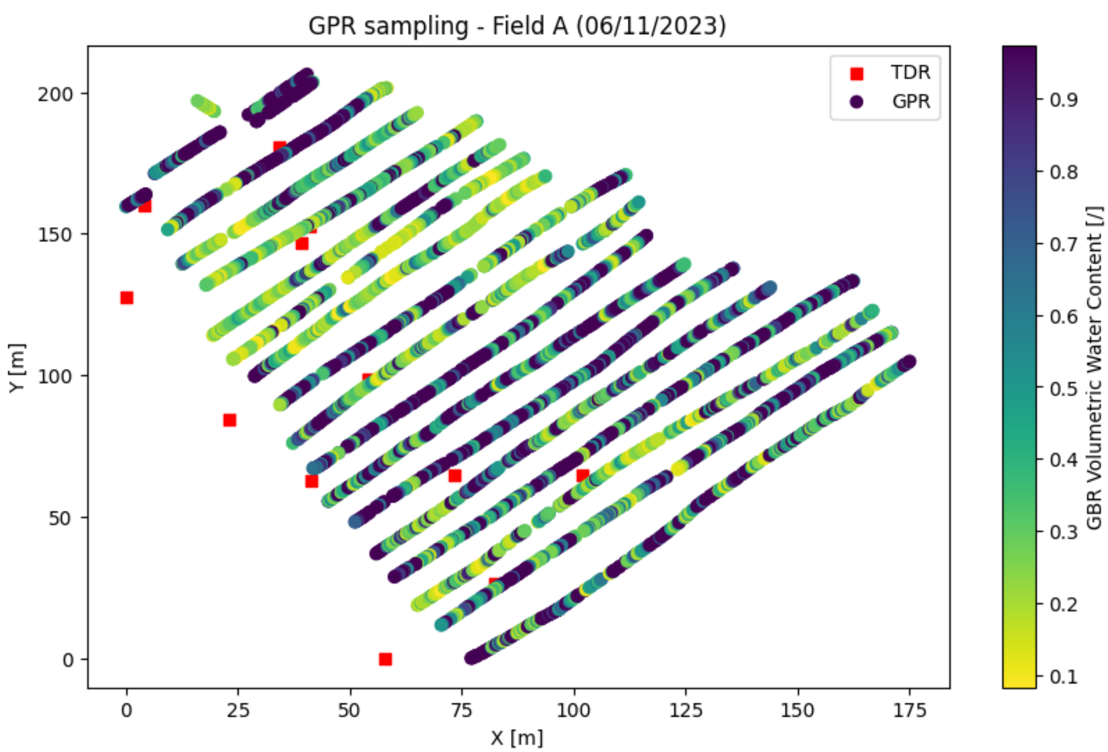



### 1.3 TDR Verification

#### 1.3.1 Field A

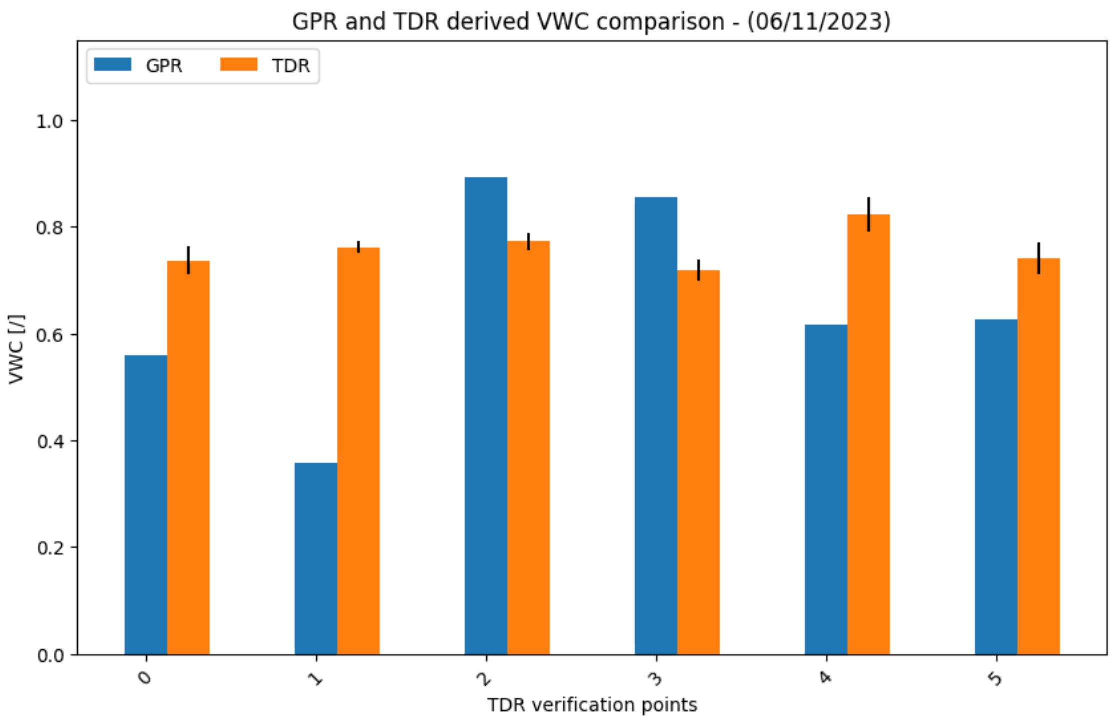
[8]:
```
GprAnalysis(field_letter="A", sample_number=8).tdr_verification(10)
```

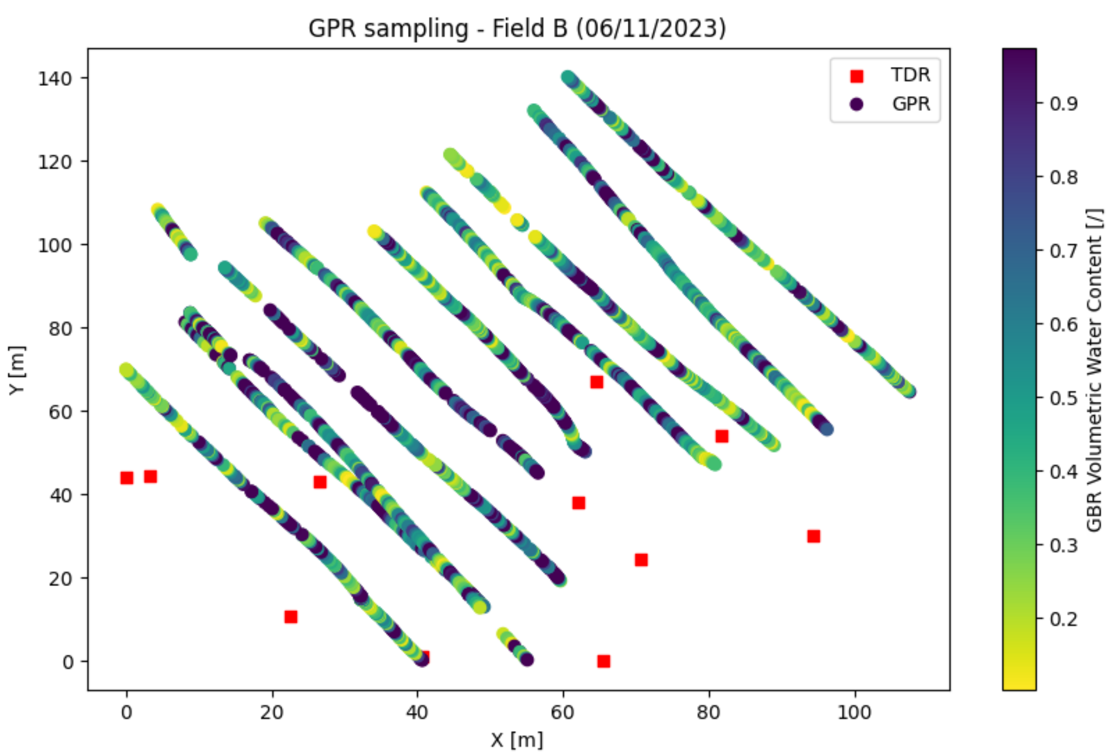

GPR and TDR derived VWC comparison - (06/11/2023)
GPR
TDR
VWC [/]
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
TDR verification points


GPR sampling - Field A (06/11/2023)
TDR
GPR
Y [m]
200
150
100
50
0
X [m]
0
25
50
75
100
125
150
175
GBR Volumetric Water Content [/]
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1

### 1.3.2 Field B

[9]:
```
GprAnalysis(field_letter="B", sample_number=8).tdr_verification(10)
```

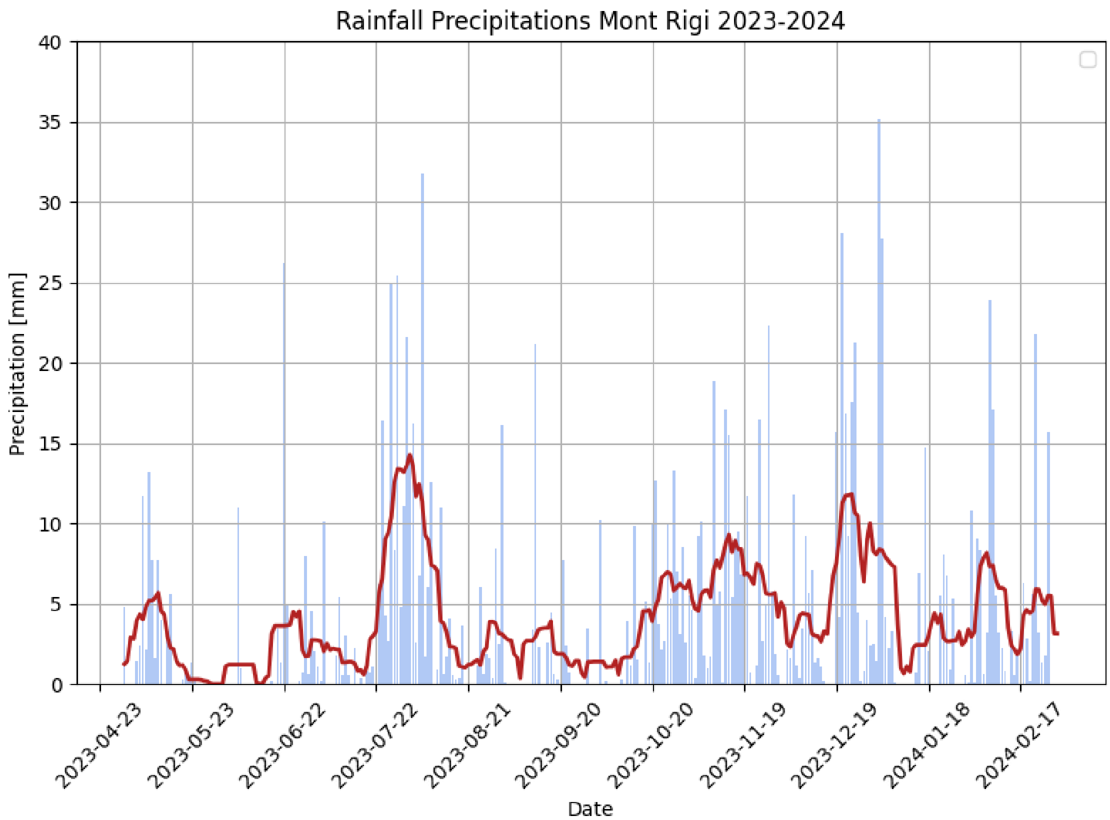


## 1.4 Rainfall Analysis

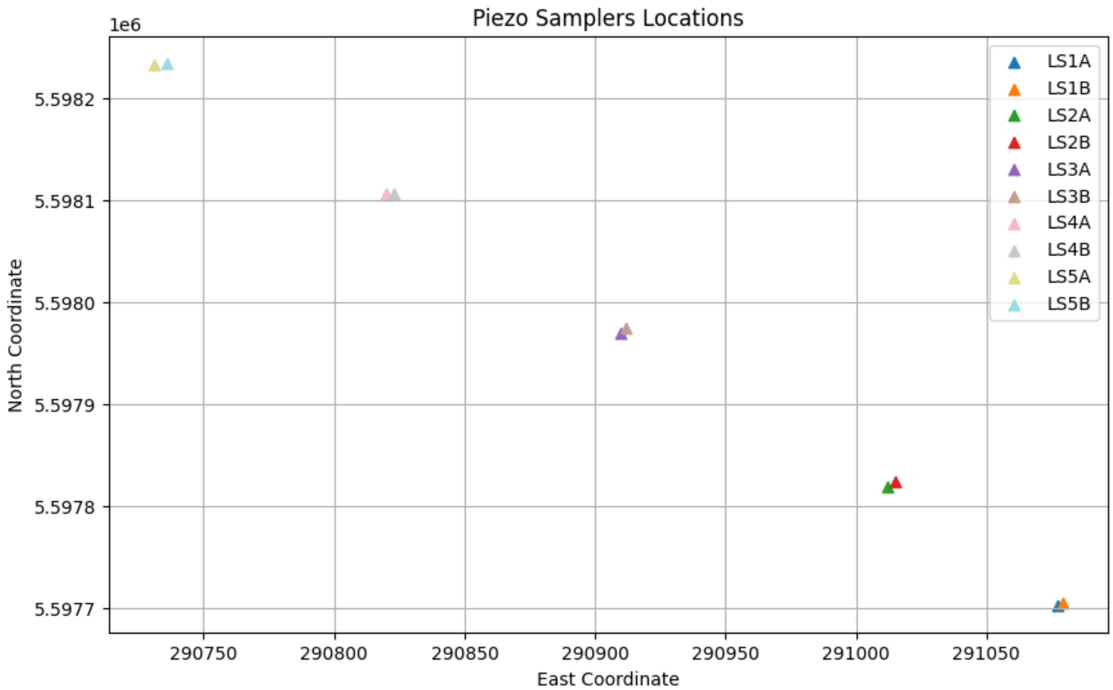
[10]:
```
rf_mr = Rainfall()
rf_mr.plot_data()
```

```
d:\Coding\Python\Master-Thesis\classes.py:565: UserWarning: No artists with
labels found to put in legend.  Note that artists whose label start with an
underscore are ignored when legend() is called with no argument.
  ax.legend()
```

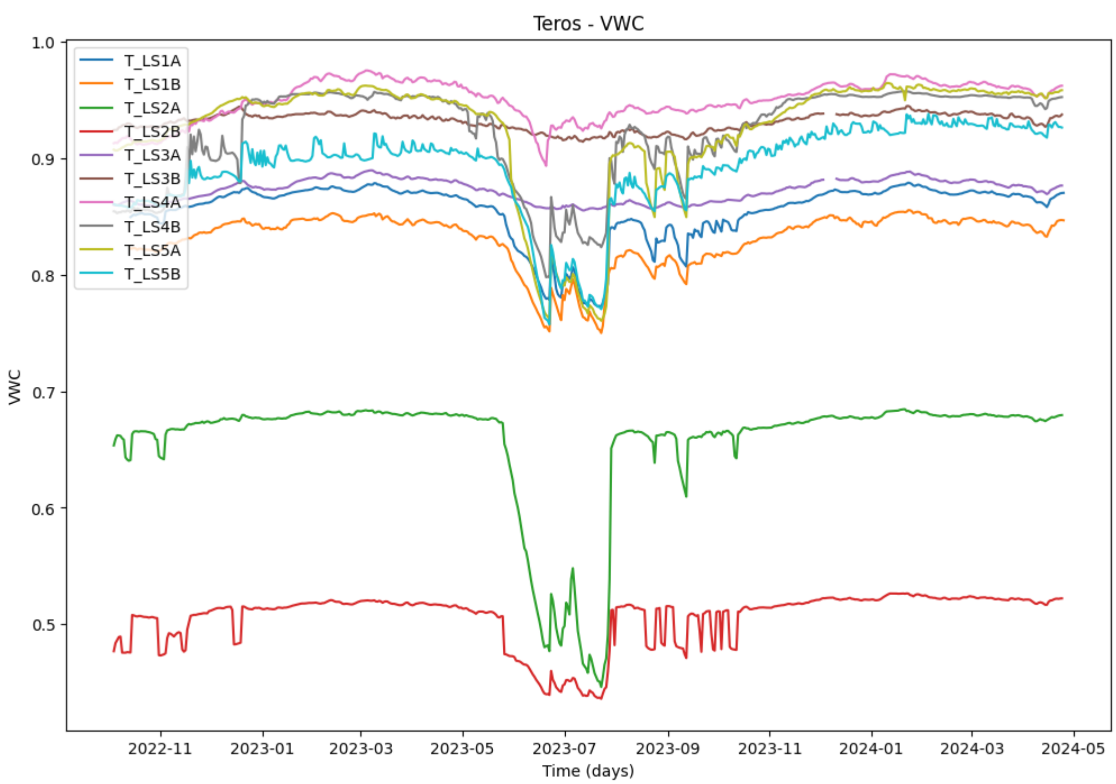



## 1.5 VWC continuous Analysis

[11]:
```
teros = Teros()
teros.plot_piezo_sampler_locations()
```

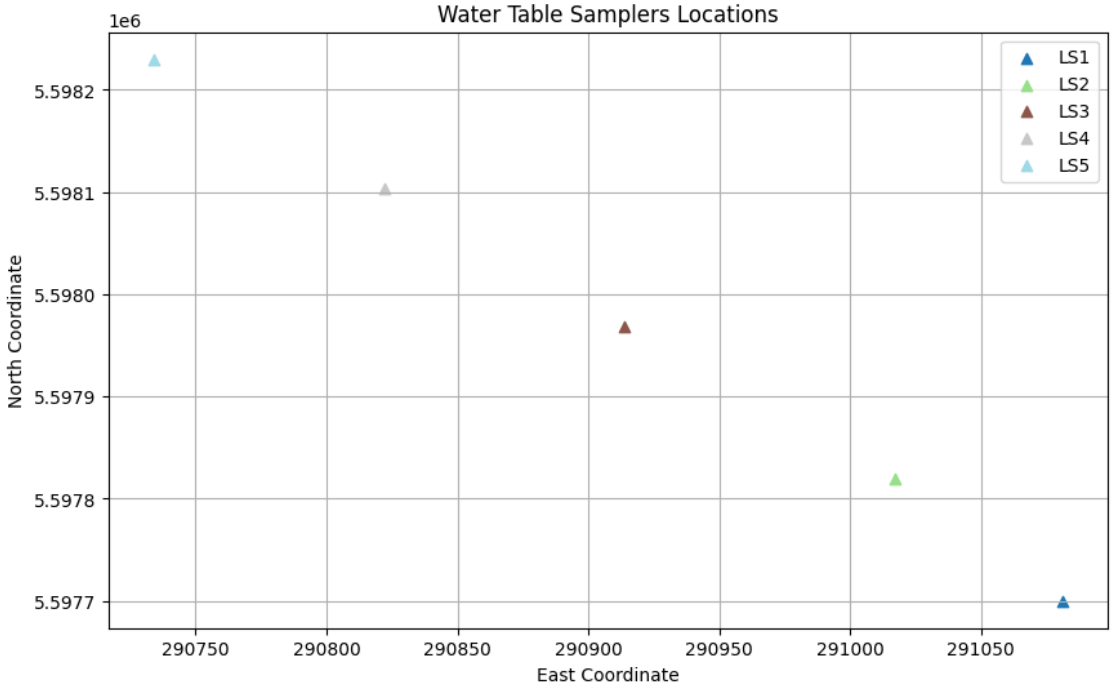



[12]:
```
teros.plot_vwc_evolution()
```

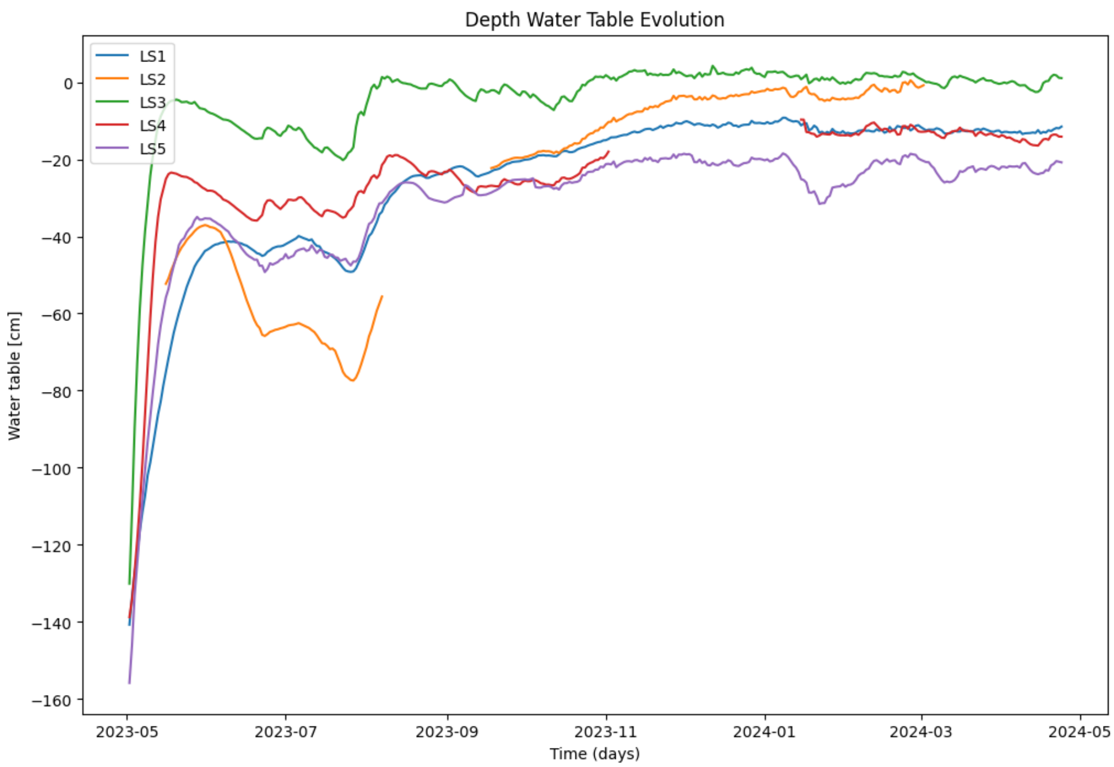


## 1.6 Water table depth Analysis

[13]:
```
wt = WaterTable()
wt.plot_wt_sampler_locations()
```




[14]:
```
wt.plot_wt_evolution()
```




## 1.7 Multispectral analysis

### 1.7.1 TVDI

The formula used for the Temperature Vegetation Dryness Index (TVDI) calculation is:

$$\text{TVDI} = \frac{\text{LST} - T_{\text{min}}(\text{NDVI})}{T_{\text{max}}(\text{NDVI}) - T_{\text{min}}(\text{NDVI})}$$

Where: - LST is the Land Surface Temperature for a given pixel. - T max NDVI is the maximum temperature for a given NDVI value, typically represented as a linear function:

$$T_{\text{max}}(\text{NDVI}) = a \cdot \text{NDVI} + b$$

- T min NDVI is the minimum temperature for a given NDVI value, typically represented as a linear function:

$$T_{\text{min}}(\text{NDVI}) = c \cdot \text{NDVI} + d$$

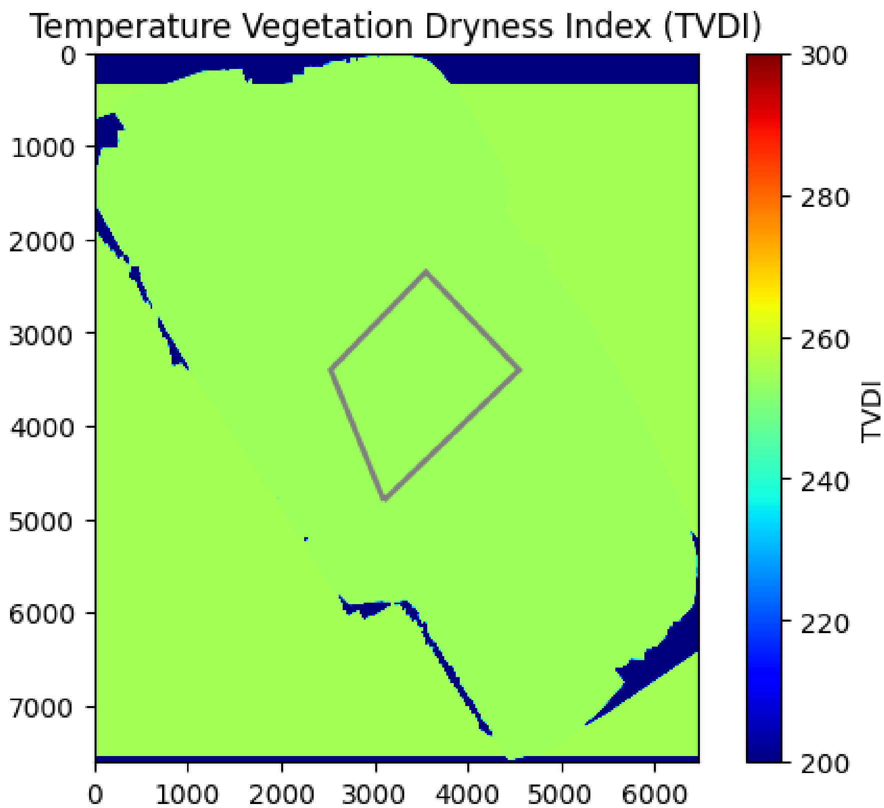
Here the specific linear functions used were:

$$T_{\text{max}}(\text{NDVI}) = 40 \cdot \text{NDVI} + 300$$

$$T_{\text{min}}(\text{NDVI}) = 20 \cdot \text{NDVI} + 250$$

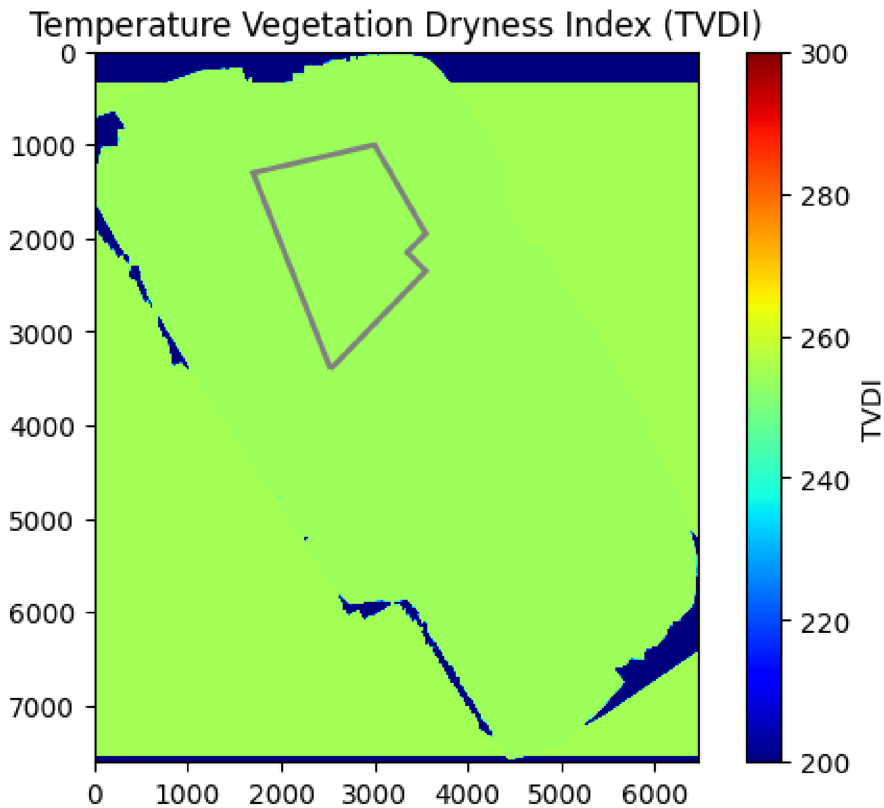
[15]:
```
multi_a = MultispecAnalysis(field_letter="A")

multi_a.calculate_tvdi()
```




[16]:
```
multi_b = MultispecAnalysis(field_letter="B")

multi_b.calculate_tvdi()
```


Temperature Vegetation Dryness Index (TVDI)
0
1000
2000
3000
4000
5000
6000
7000
0
1000
2000
3000
4000
5000
6000
300
280
260
240
220
200
TVDI