Project 1: data fusion with change of support

Package, Simulation, Sampling

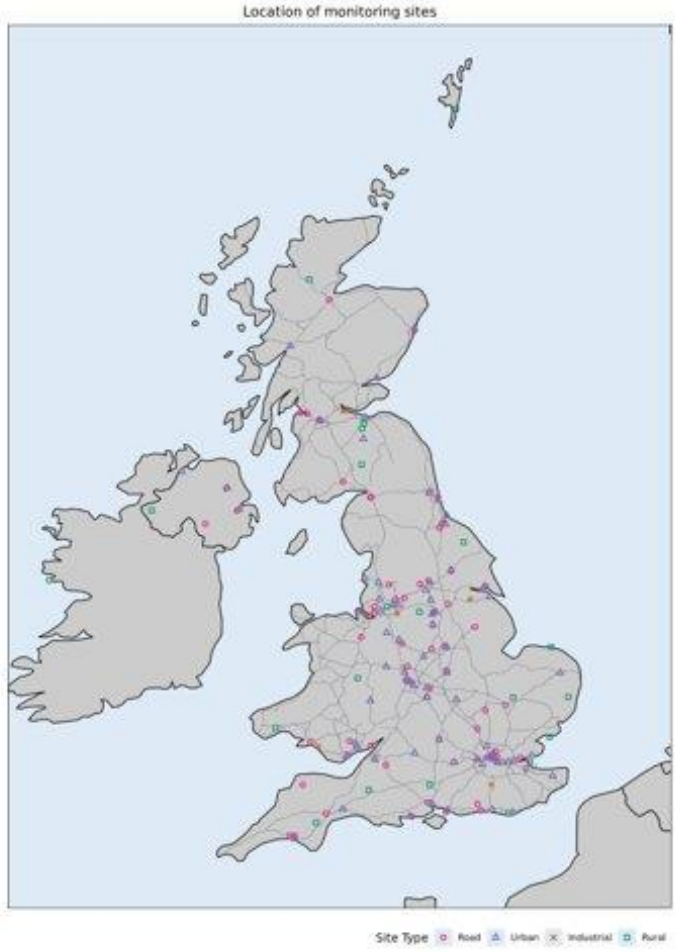
Ana Couto[1], Fergus Chadwick[1], David Miller[1,2], Jackie Potts[1], Thomas Cornulier[1]

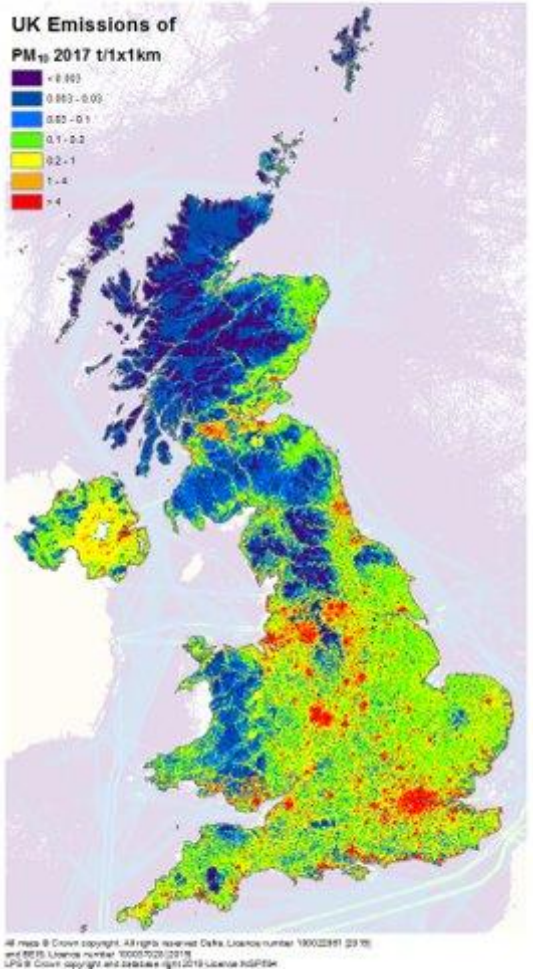
1 BioSS

2 UK Centre for Ecology & Hydrology

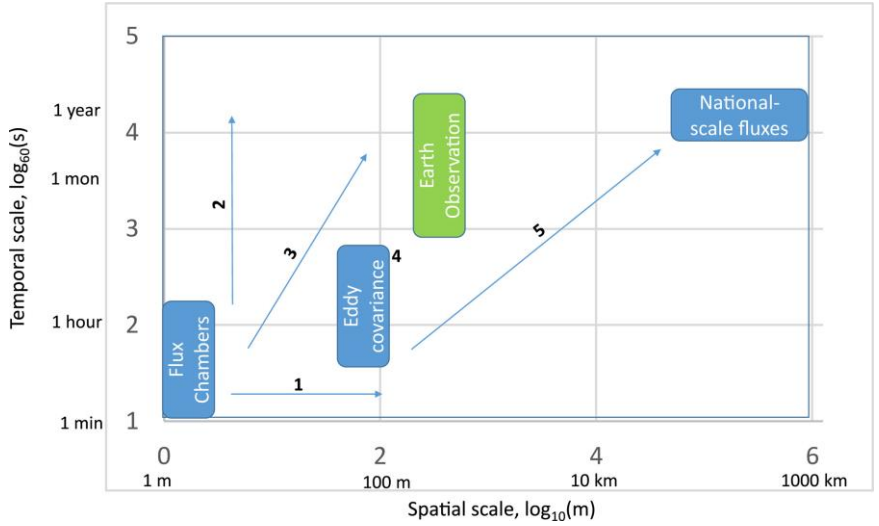
Case studies

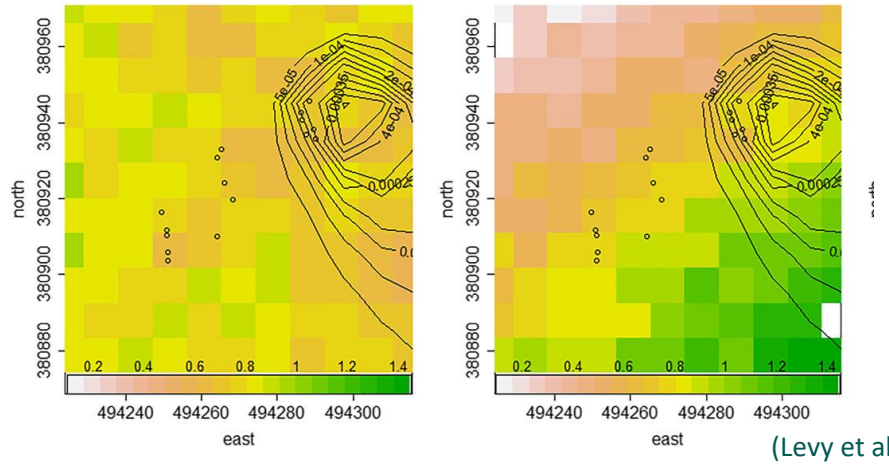
Atmospheric pollutants





Greenhouse Gas Fluxes

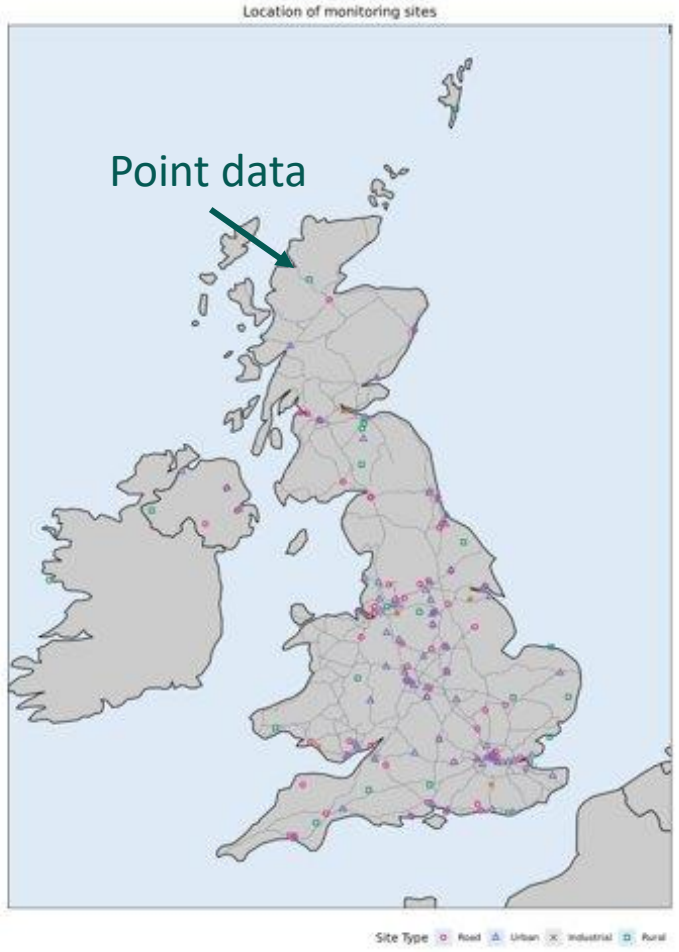


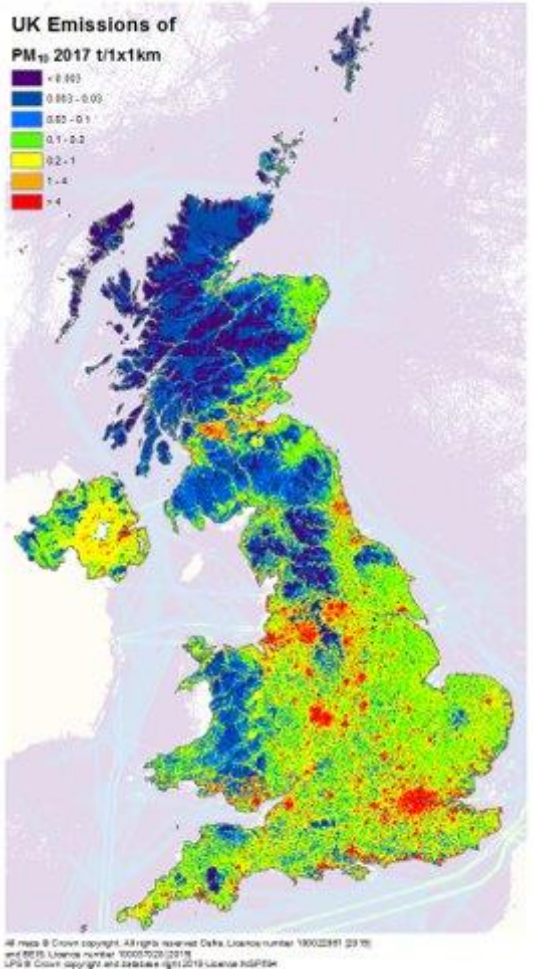


(Levy et al., 2022)

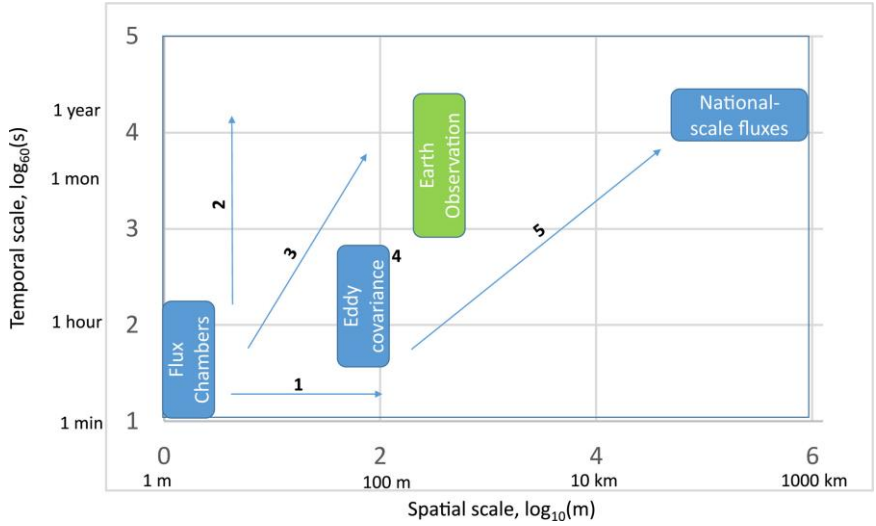
Case studies

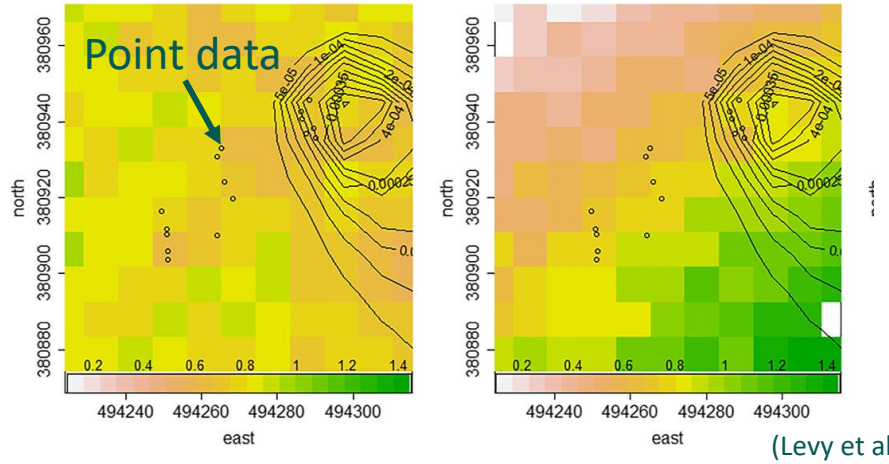
Atmospheric pollutants





Greenhouse Gas Fluxes

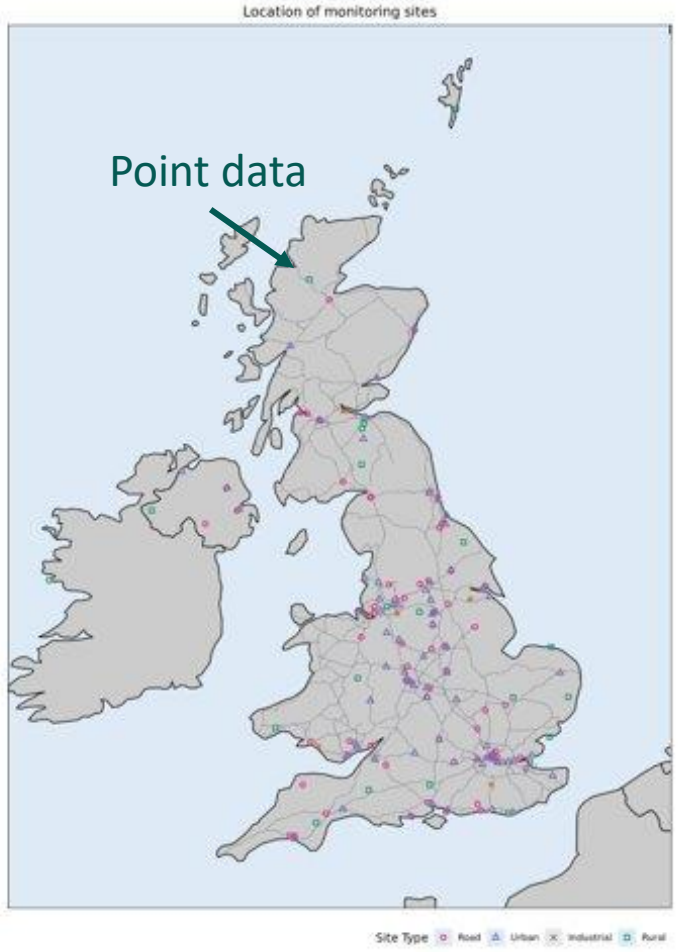


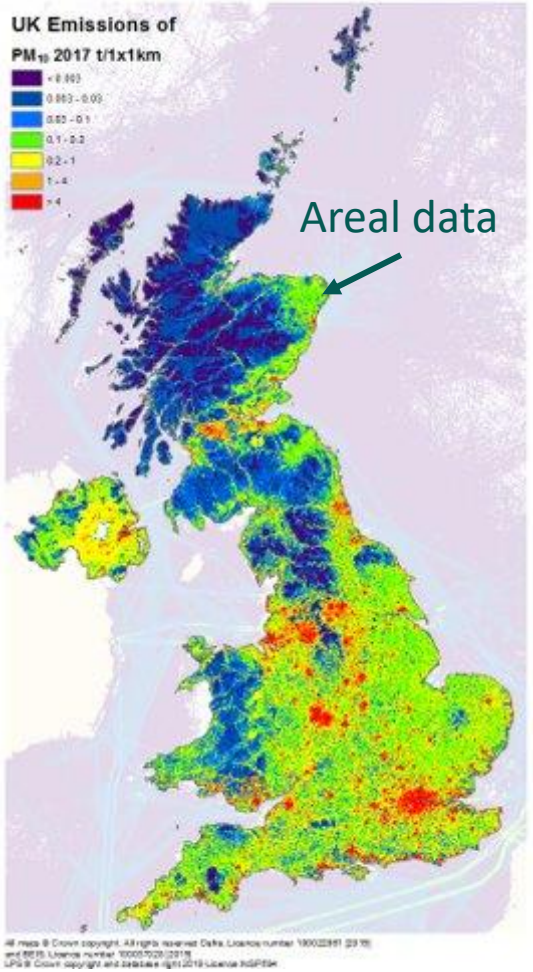


(Levy et all., 2022)

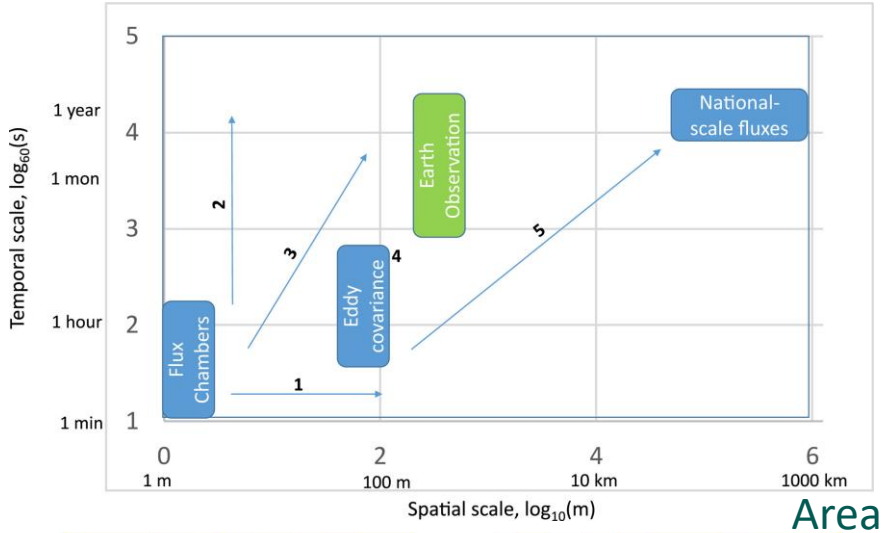
Case studies

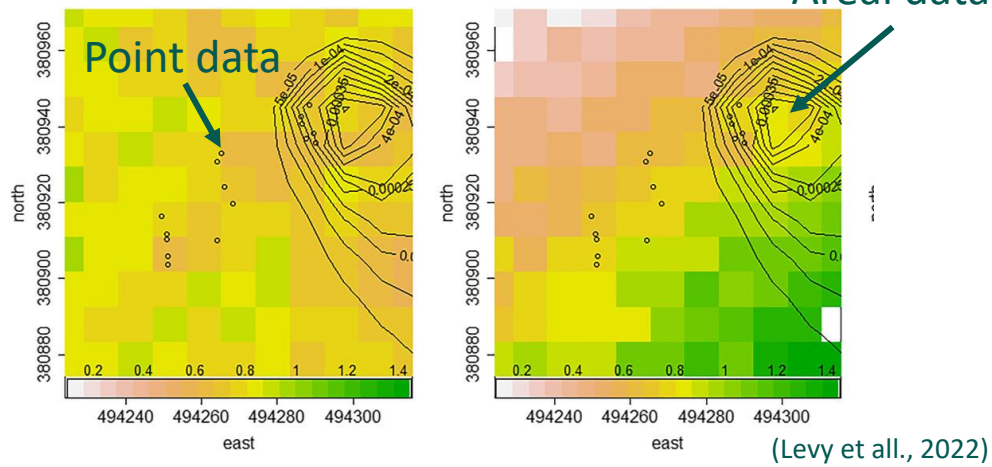
Atmospheric pollutants





Greenhouse Gas Fluxes





(Levy et all., 2022)

Generalising the case studies

- Two data sources for each underlying process – point and area
- (Spatio-)temporal smoothness
- Sparse point sampling
- No gold-standard/Truth™.

Generalising the case studies

- Two data sources for each underlying process – point and area
- (Spatio-)temporal smoothness
- Sparse point sampling
- No gold-standard/Truth™.

What we did

- Wrote a software package to simulate a wide range of systems and sampling strategies (including air pollution and greenhouse gas patterns);
- Provided a suite of methods for integrating and comparing sparse spacetime data sources;
- Developing guidelines for interrogating and interpreting these models.

What we did

- **Wrote a software package to simulate a wide range of systems and sampling strategies (including air pollution and greenhouse gas patterns);**
- Provided a suite of methods for integrating and comparing sparse spacetime data sources;
- Developing guidelines for interrogating and interpreting these models.

ASCOT R package

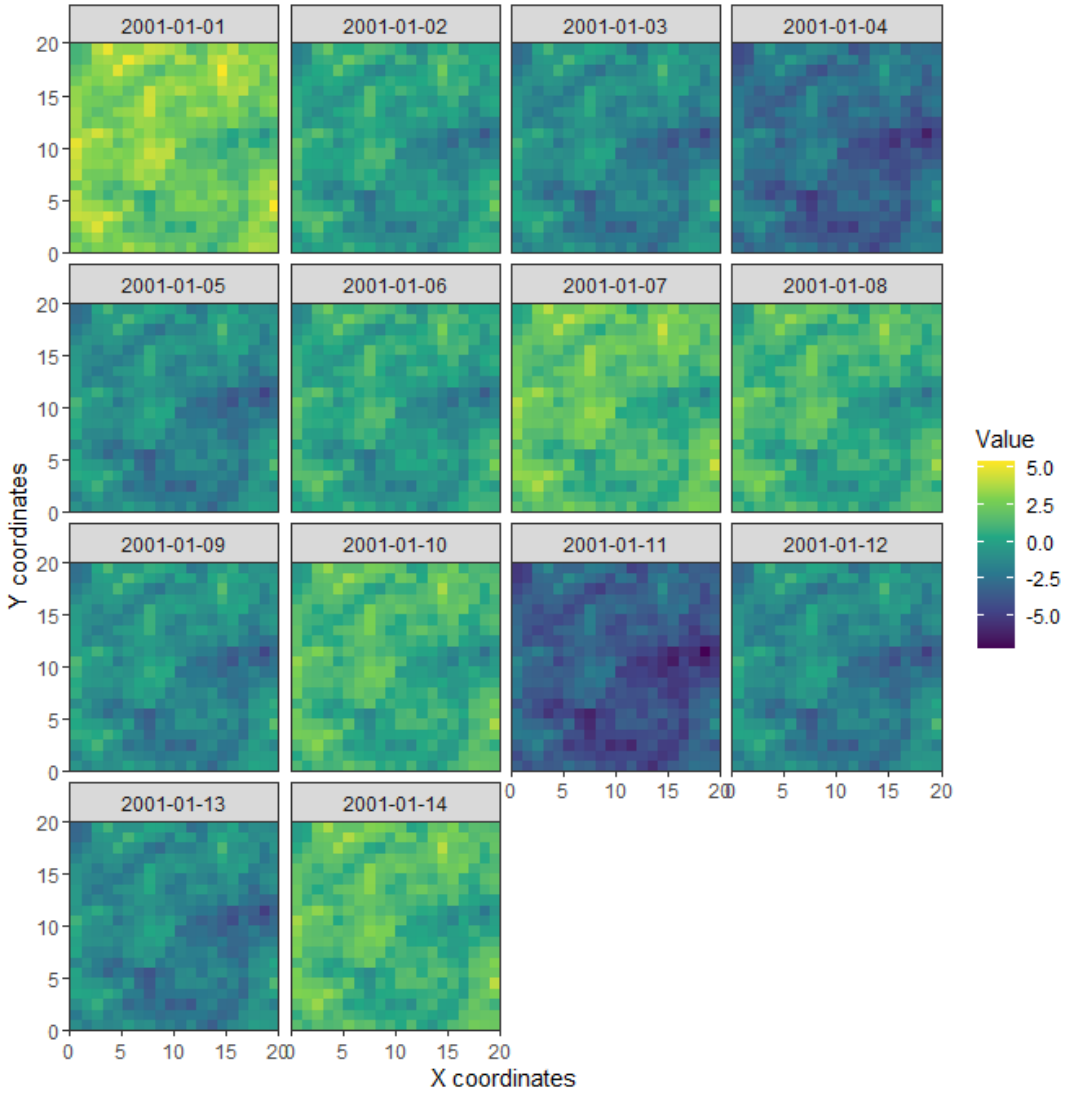
By simulating the data, you know the "truth" behind the system. This can be useful to:

- Check model performance under one or more scenarios
- Check number of samples required

```
install.packages("devtools")
devtools::install_git("https://gitlab.bioss.ac.uk/ukceh-bioss-framework/dev/ascot.git", dependencies = TRUE )
```

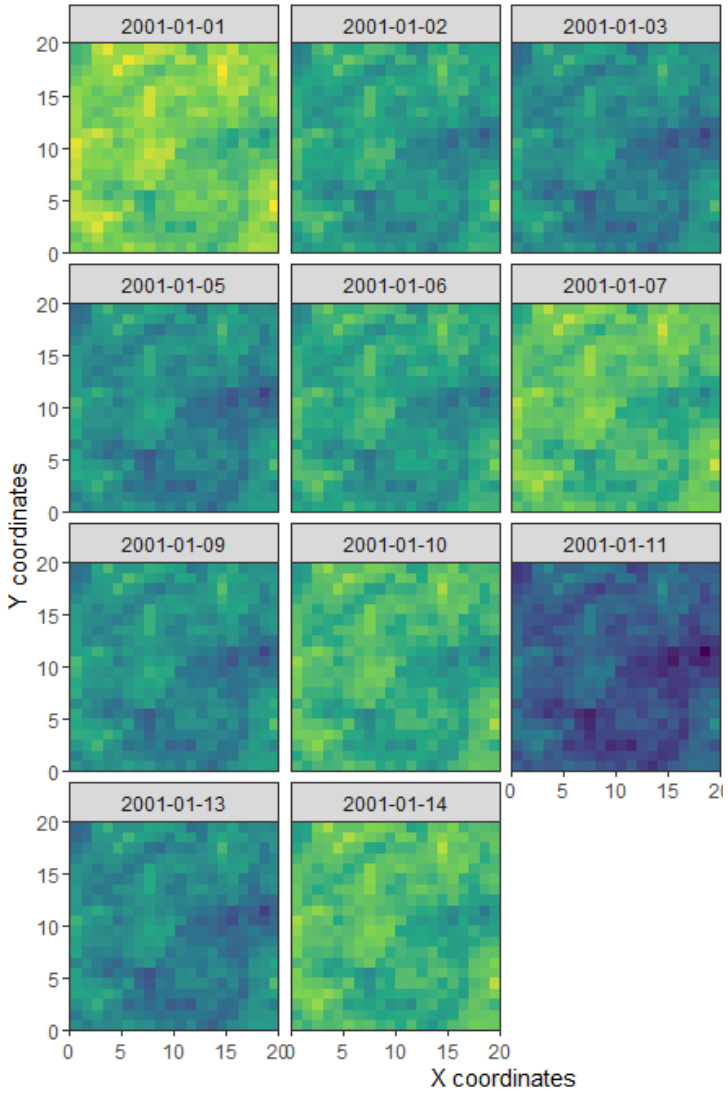
Simulating the “truth”

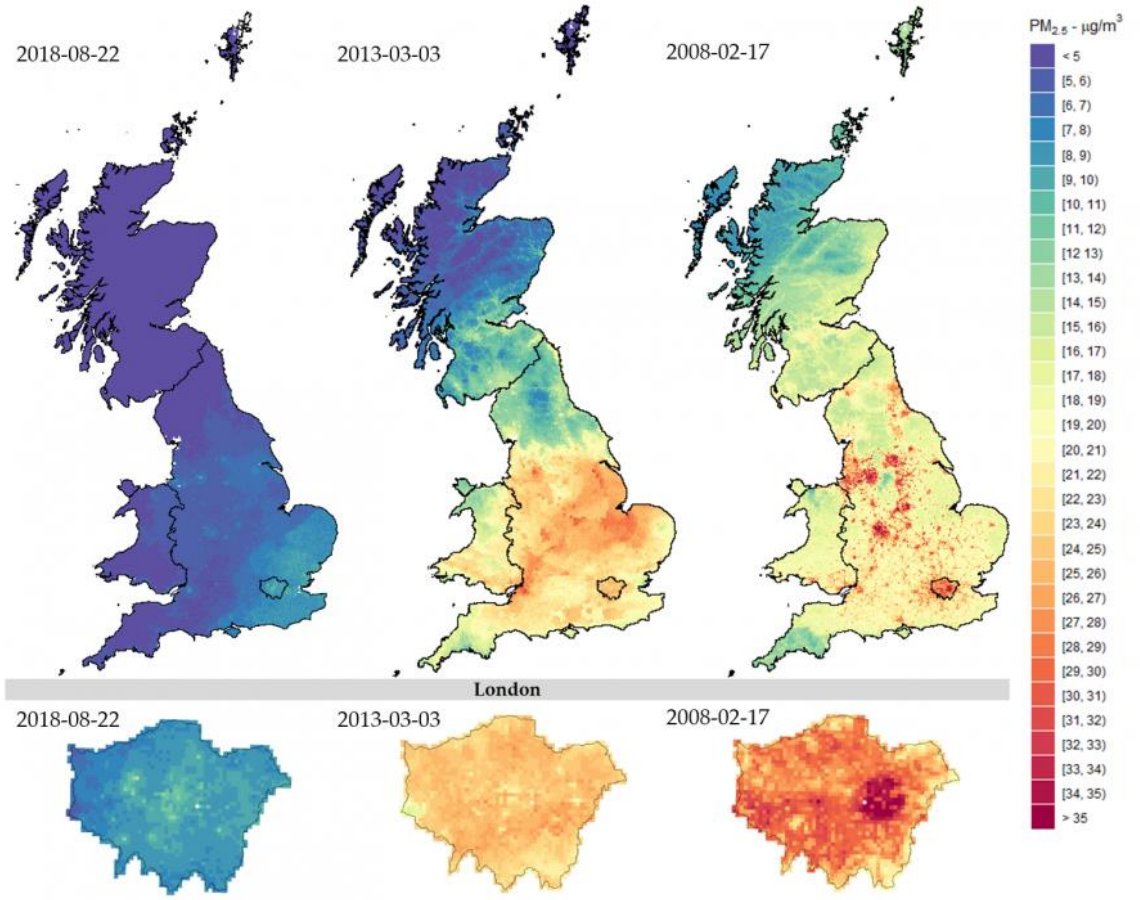
- Temporal component
- Spatial component



Simulating the “truth”

- Temporal component
- Spatial component

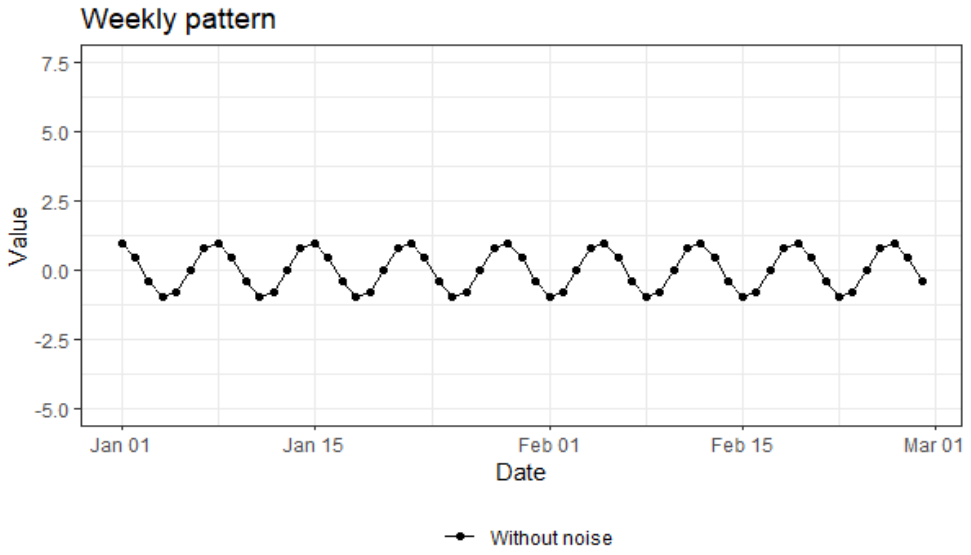




Day specific PM2.5 estimations across Great Britain (top) and London (bottom) (Schneider et al., 2020)

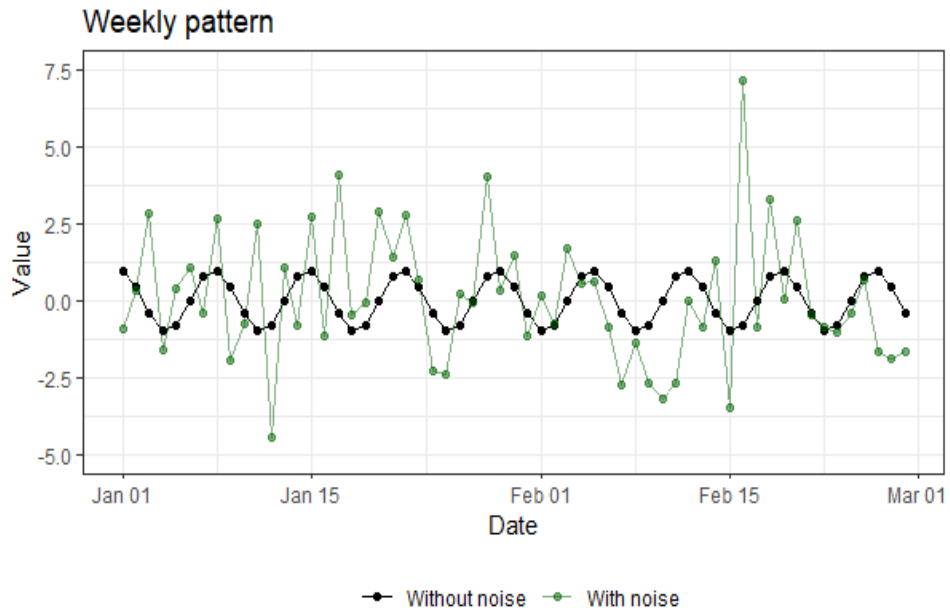
Simulate temporal data

```
# Create a time series
time_period <-set_timeline(2, "month")

# Set week pattern without noise
time_series<- define_pattern_t(time_series= time_period ,
                               weekly_amplitude = 1,
                               weekly_noise = 0,
                               seasonal_amplitude = 0,
                               seasonal_noise = 0,
                               annual_amplitude = 0,
                               annual_noise = 0)
```



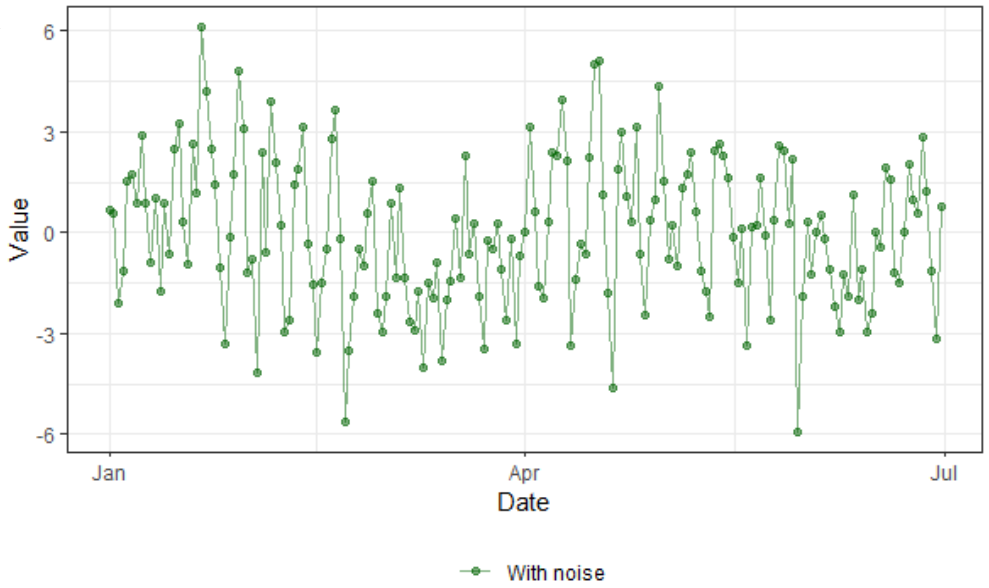
Simulate temporal data

```
# Create a time series
time_period<-set_timeline(2, "month")

# Set week pattern without noise
time_series<- define_pattern_t(time_series= weekly_time,
                               weekly_amplitude = 1,
                               weekly_noise = 2,
                               seasonal_amplitude = 0,
                               seasonal_noise = 0,
                               annual_amplitude = 0,
                               annual_noise = 0)
```



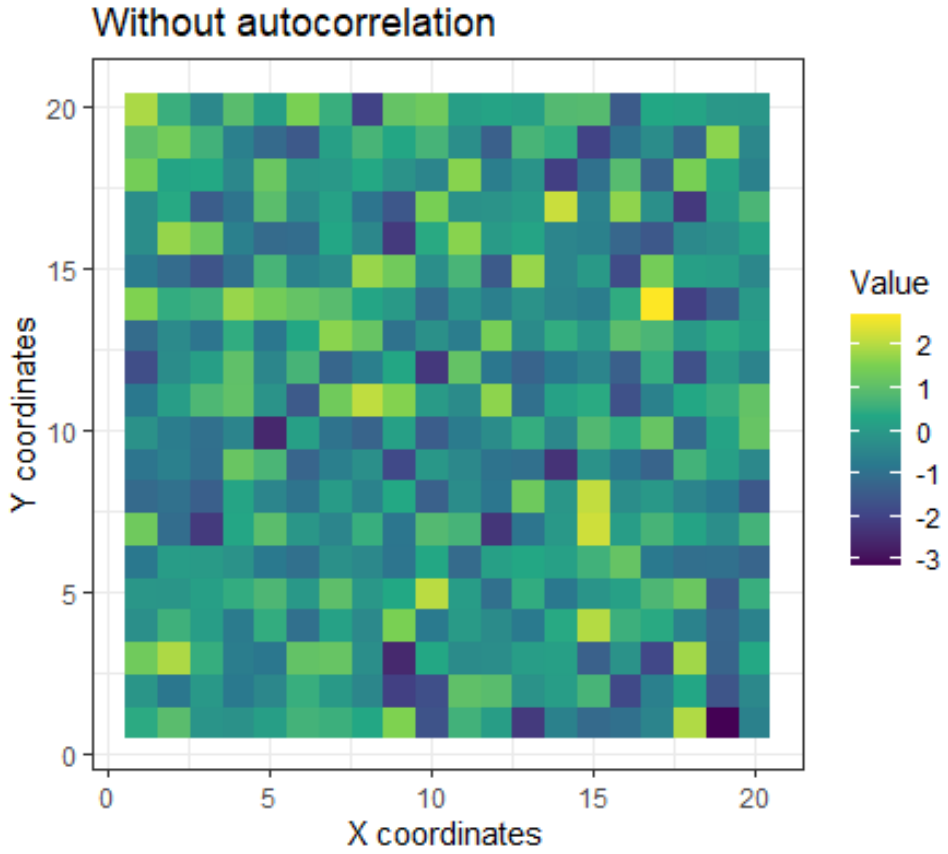
Simulate temporal data

```r
# Create a time series
time_period<-set_timeline(6, "month")

# Set week pattern without noise
time_series <- define_pattern_t(time_series= time_period,
                                weekly_amplitude = 1,
                                weekly_noise = 2,
                                seasonal_amplitude = 1,
                                seasonal_noise = 1,
                                annual_amplitude = 0,
                                annual_noise = 0)
```



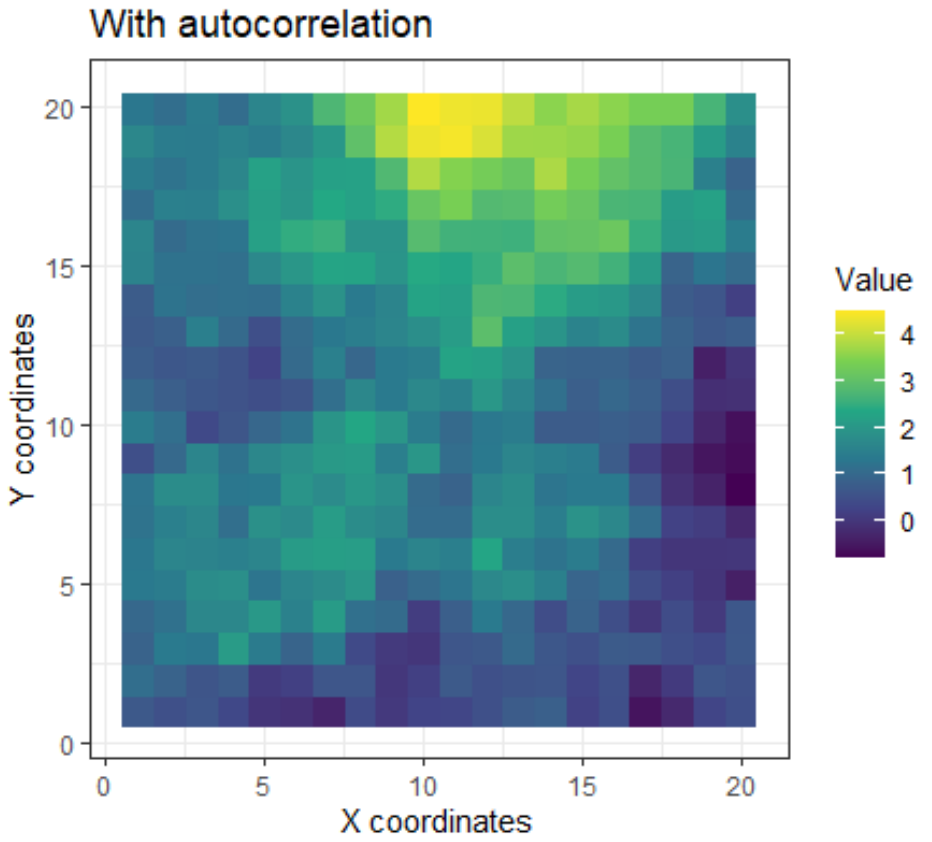
Simulate spatial data

```
# Create spatial field
spatial_field <- define_pattern_s(n_x = 20,
                                  n_y = 20,
                                  autocorr = FALSE)
```



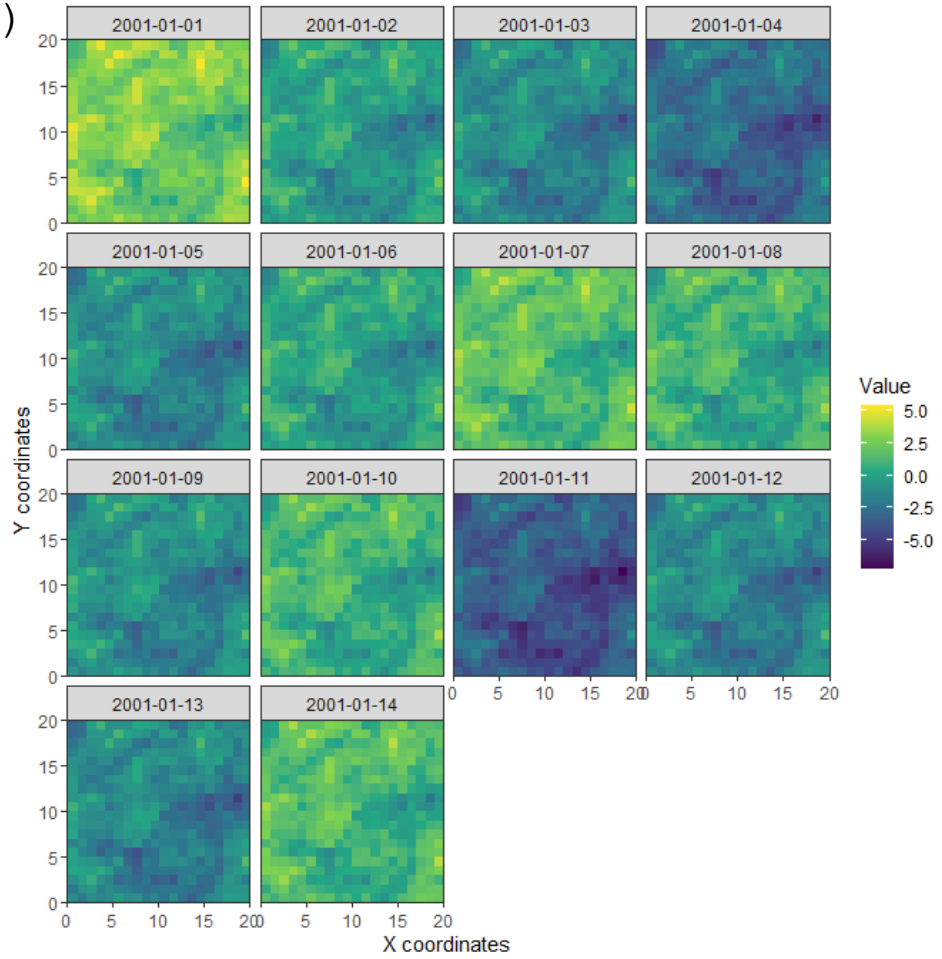
Simulate spatial data

```
# Create spatial field
spatial_field <- define_pattern_s(n_x = 20,
                                  n_y = 20,
                                  autocorr = TRUE,
                                  range = 10,
                                  amplitude = 1)
```



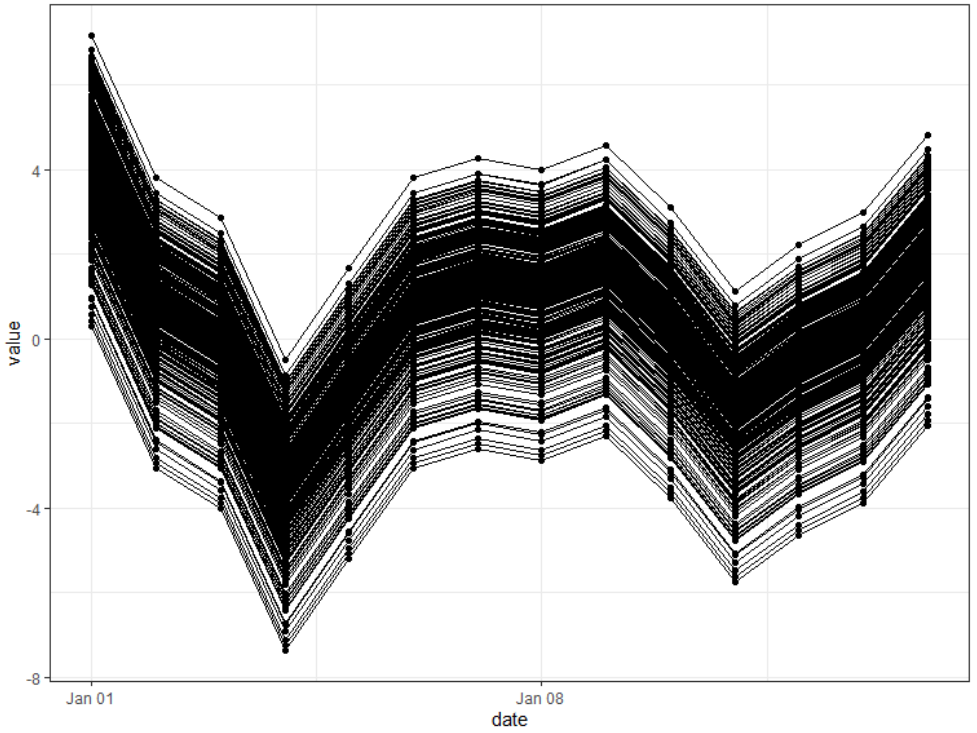
Simulate spatio-temporal data

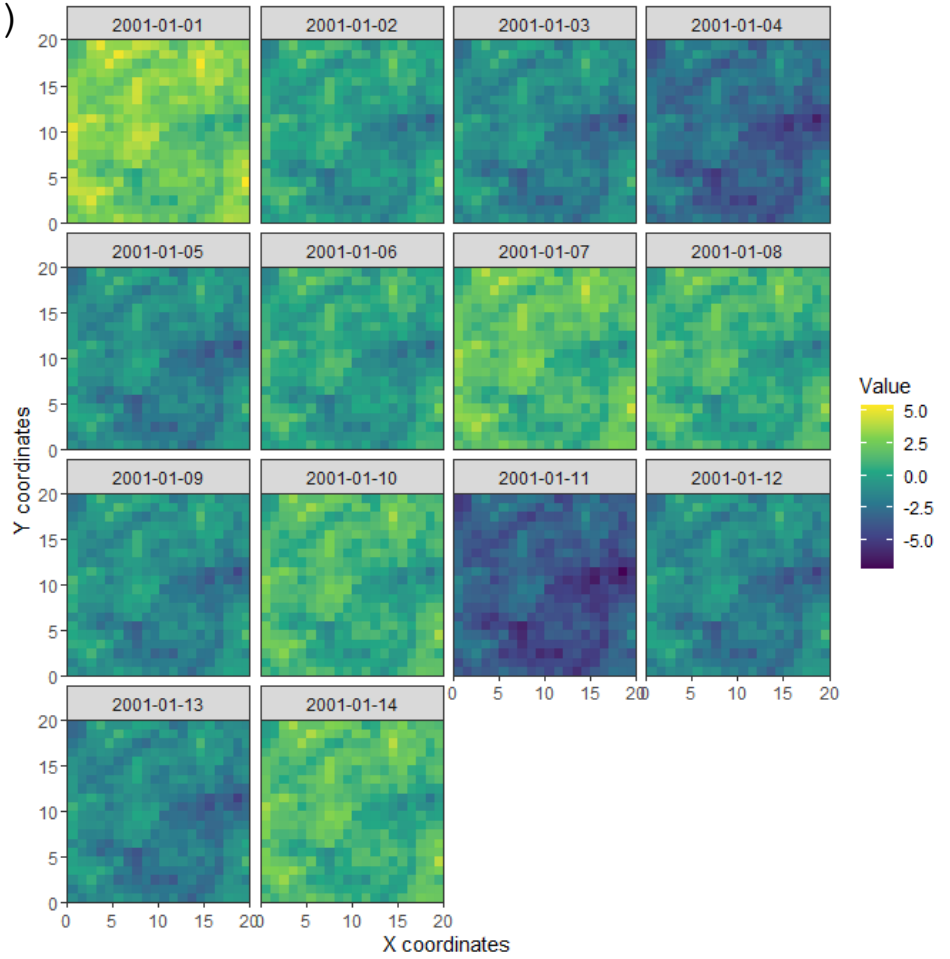
```
truth <- simulate_spacetime(temporal_pattern = time_series,
                            field = spatial_field,
                            pattern_adherence = 0)
```



Simulate spatio-temporal data

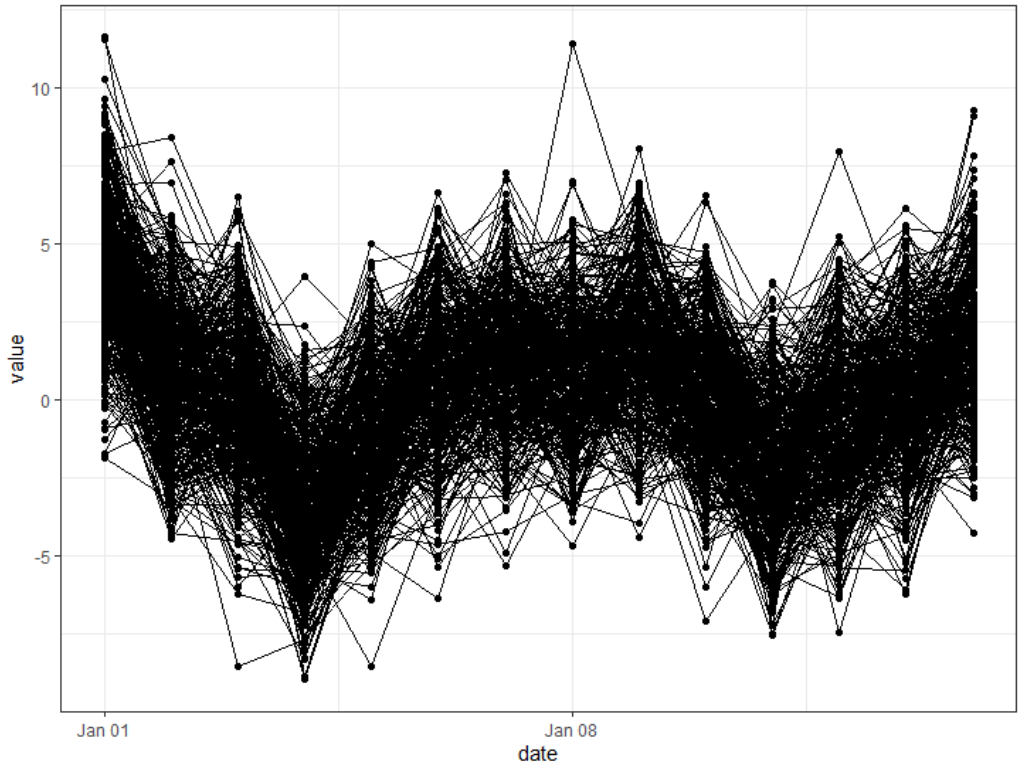
```
truth <- simulate_spacetime(temporal_pattern = time_series,
                            field = spatial_field,
                            pattern_adherence = 0)
```

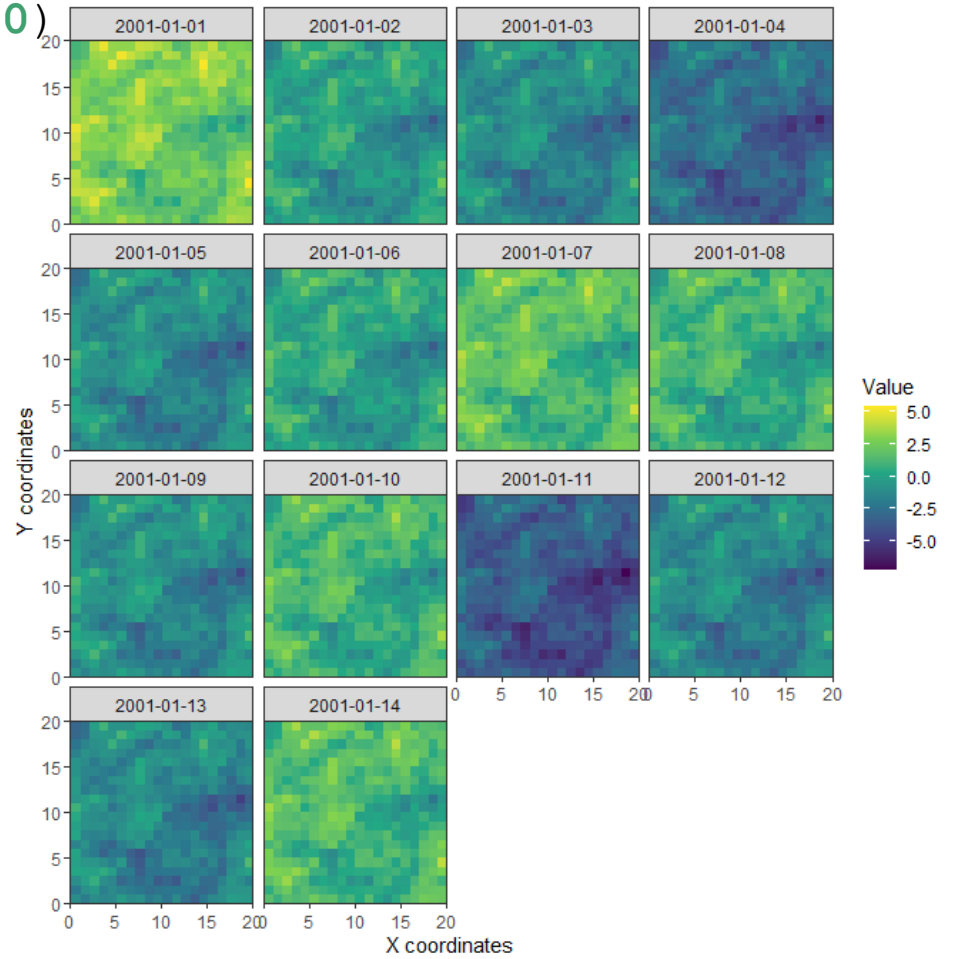




Simulate spatio-temporal data

```
truth <- simulate_spacetime(temporal_pattern = time_series,
                            field = spatial_field,
                            pattern_adherence = 10)
```

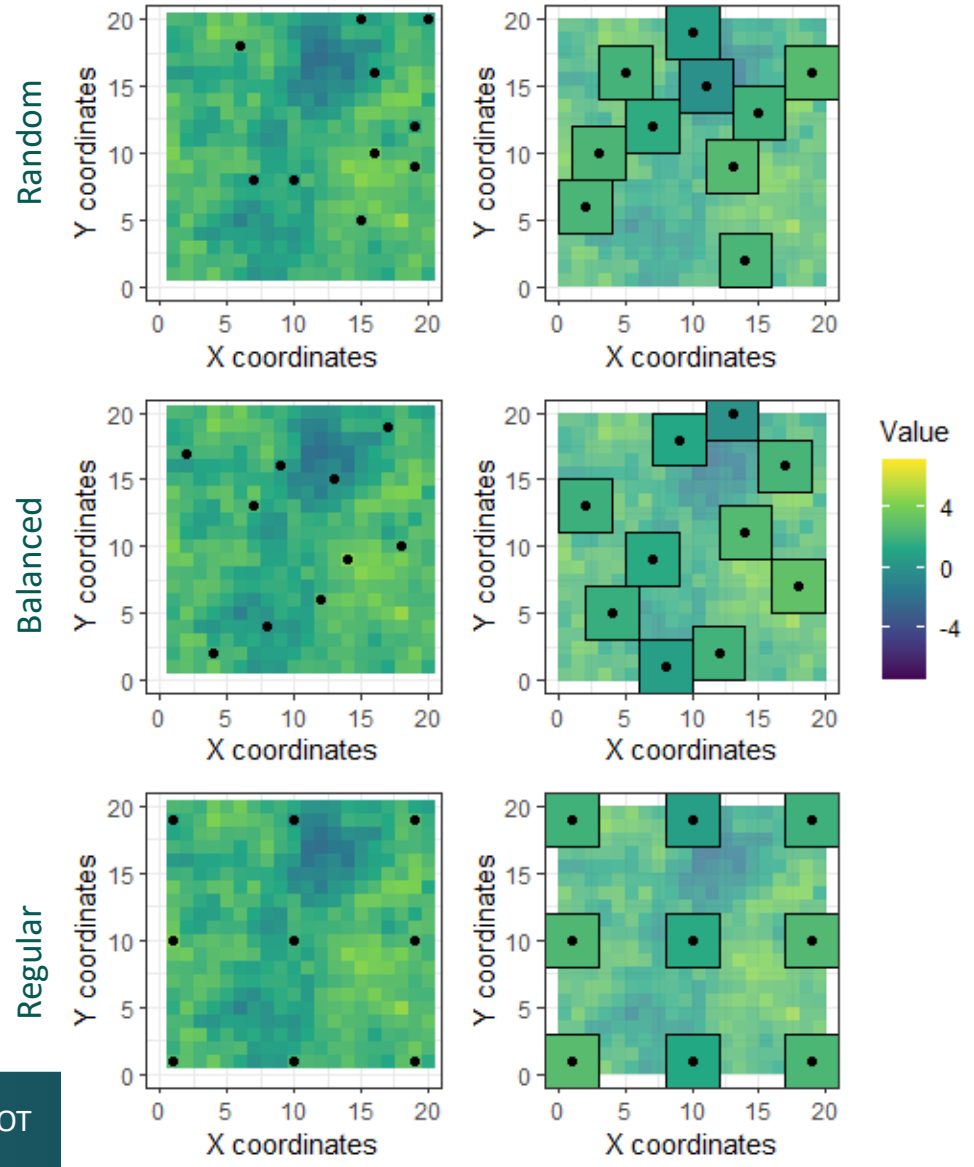




Sampling from the “truth”

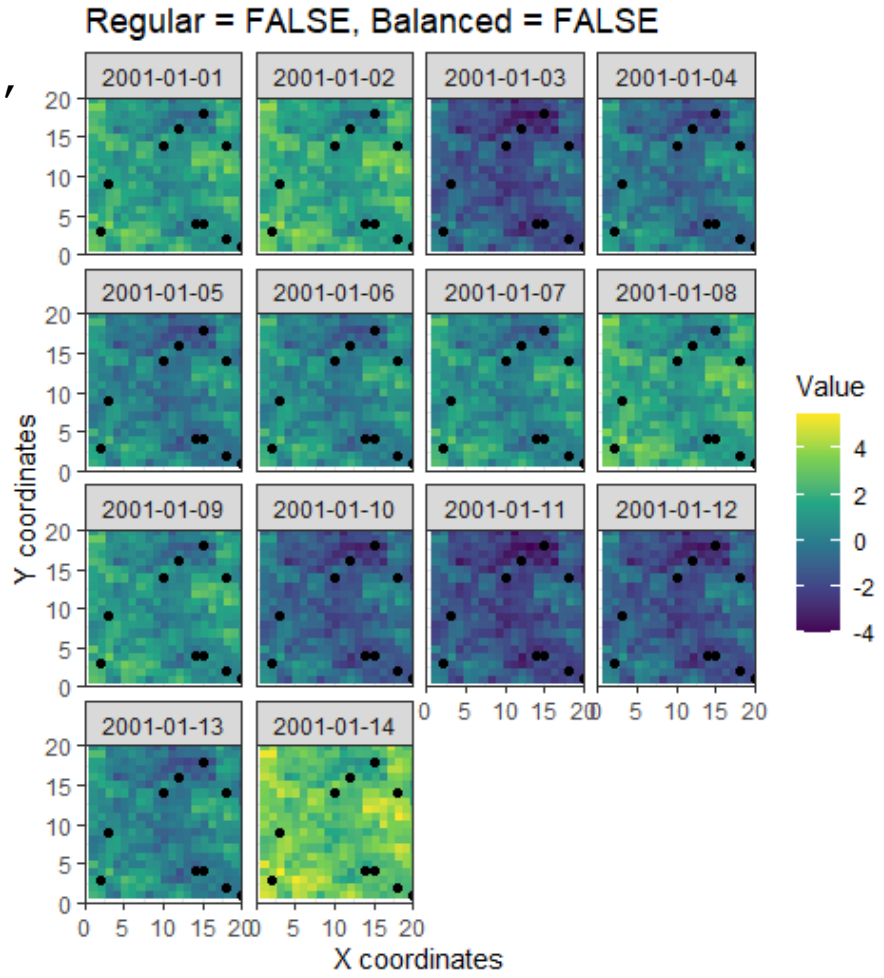
Three types of sampling designs

1. Random sampling where pixels are randomly selected
2. A "spatially balanced" design, which changes the inclusion probabilities to try to avoid clustering of samples
3. A regular grid



Point sampling

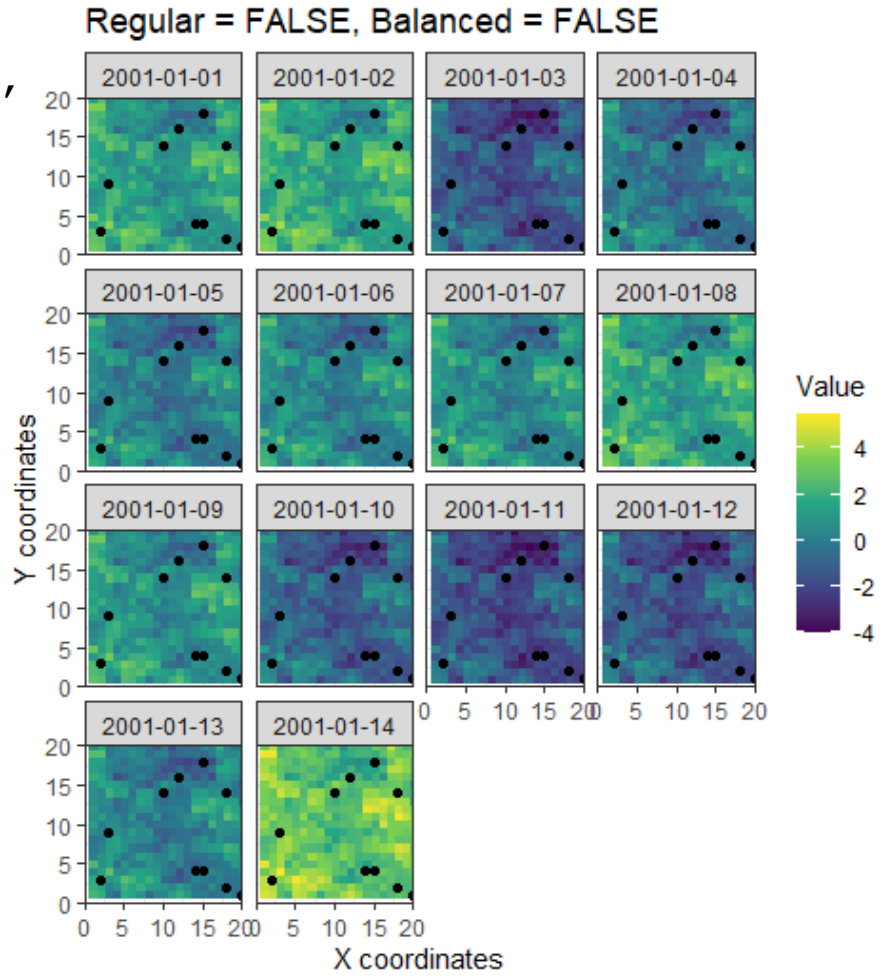
```
#random sampling
point_sample <- sample_area_pointily(field= truth,
                                     n_samples=10,
                                     sampling_error=0.1,
                                     regular = FALSE,
                                     balanced = FALSE)
```



Point sampling

```
#random sampling
point_sample <- sample_area_pointily(field= truth,
                                     n_samples=10,
                                     sampling_error=0.1,
                                     regular = FALSE,
                                     balanced = FALSE)
```

x	y	date	value	point_sample
14	4	2001-01-01	0.859994152	1.01451041
14	4	2001-01-02	1.308366898	1.25523819
14	4	2001-01-03	-1.887442465	-1.92402065
14	4	2001-01-04	-0.724117975	-0.61466940
14	4	2001-01-05	-0.351386711	-0.41457173
14	4	2001-01-06	-0.023788588	-0.03178429
14	4	2001-01-07	0.248962961	0.09181252
14	4	2001-01-08	1.097373716	1.11349034
14	4	2001-01-09	0.484955174	0.58645841
14	4	2001-01-10	-1.418660698	-1.36722443
14	4	2001-01-11	-2.077634433	-2.19358839
14	4	2001-01-12	-1.784752134	-1.49943629
14	4	2001-01-13	-0.135135376	-0.22487732
14	4	2001-01-14	2.571453959	2.48995735

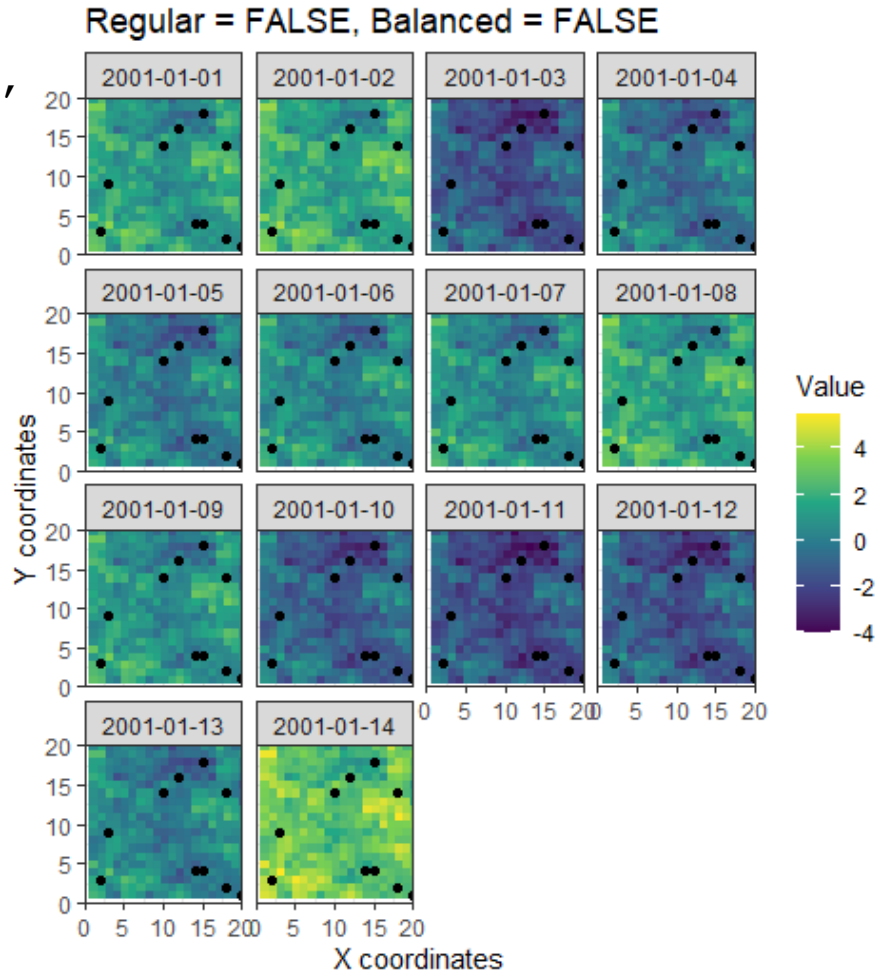


Point sampling

```
#random sampling
point_sample <- sample_area_pointily(field= truth,
                                     n_samples=10,
                                     sampling_error=0.1,
                                     regular = FALSE,
                                     balanced = FALSE)
```

x	y	date	value	point_sample
14	4	2001-01-01	0.859994152	[illegible]
14	4	2001-01-02	1.308366898	1.25523819
14	4	2001-01-03	-1.887442465	-1.92402065
14	4	2001-01-04	-0.724117975	-0.61466940
14	4	2001-01-05	-0.351386711	-0.41457173
14	4	2001-01-06	-0.023788588	-0.03178429
14	4	2001-01-07	0.248962961	0.09181252
14	4	2001-01-08	1.097373716	1.11349034
14	4	2001-01-09	0.484955174	0.58645841
14	4	2001-01-10	-1.418660698	-1.36722443
14	4	2001-01-11	-2.077634433	-2.19358839
14	4	2001-01-12	-1.784752134	-1.49943629
14	4	2001-01-13	-0.135135376	-0.22487732
14	4	2001-01-14	2.571453959	2.48995735

"True" value

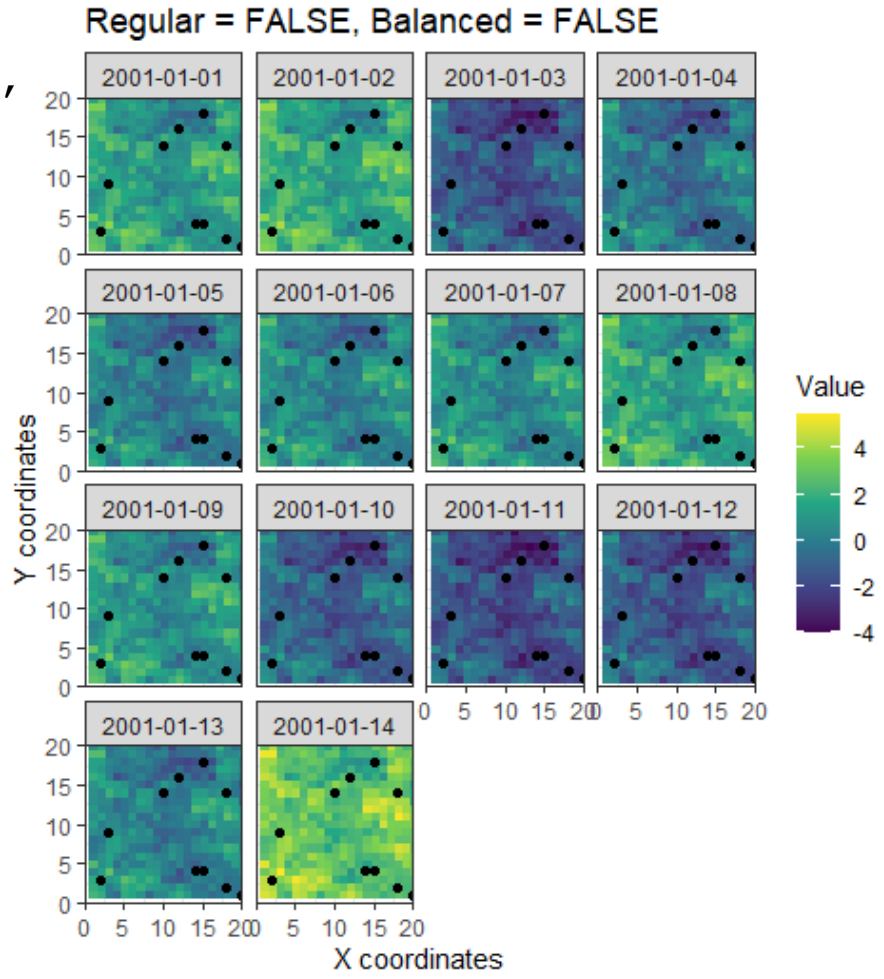


Point sampling

```
#random sampling
point_sample <- sample_area_pointily(field= truth,
                                     n_samples=10,
                                     sampling_error=0.1,
                                     regular = FALSE,
                                     balanced = FALSE)
```

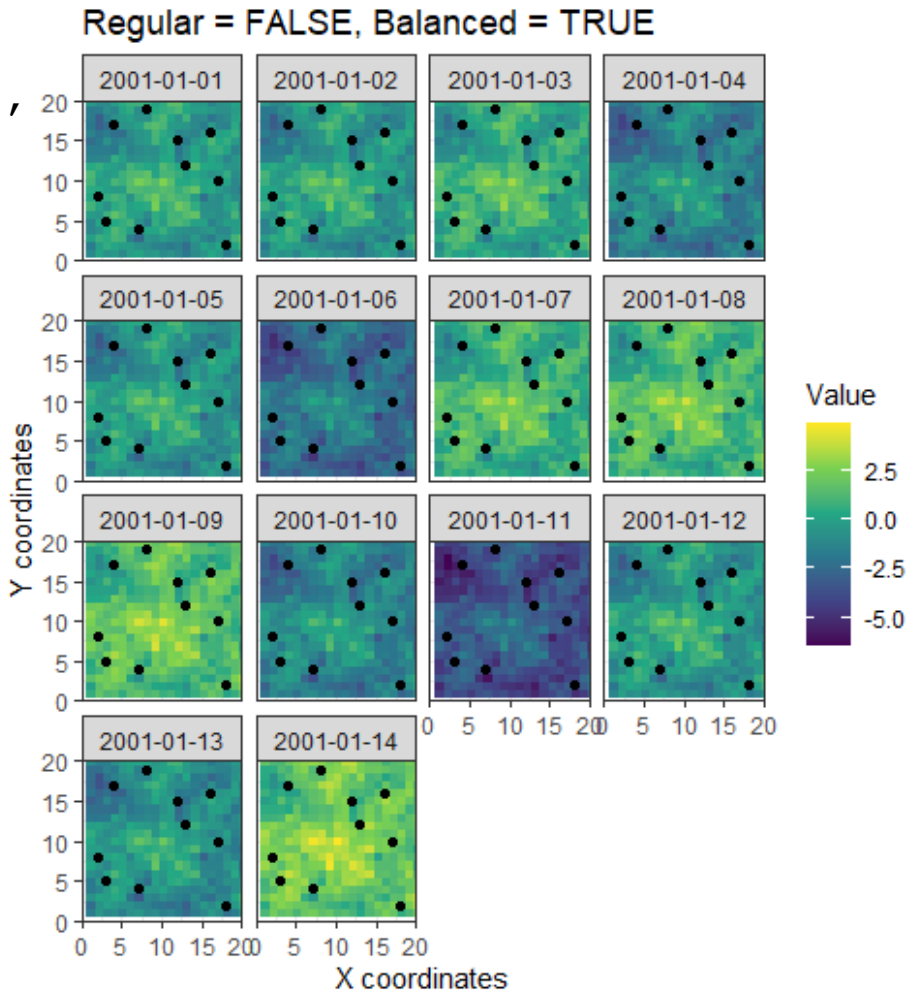
x	y	date	value	point_sample
14	4	2001-01-01	0.859994152	1.01451041
14	4	2001-01-02	1.308366898	1.25523819
14	4	2001-01-03	-1.887442465	-1.92402065
14	4	2001-01-04	-0.724117975	-0.61466940
14	4	2001-01-05	-0.351386711	-0.41457173
14	4	2001-01-06	-0.023788588	-0.03178429
14	4	2001-01-07	0.248962961	0.09181252
14	4	2001-01-08	1.097373716	1.11349034
14	4	2001-01-09	0.484955174	0.58645841
14	4	2001-01-10	-1.418660698	-1.36722443
14	4	2001-01-11	-2.077634433	-2.19358839
14	4	2001-01-12	-1.784752134	-1.49943629
14	4	2001-01-13	-0.135135376	-0.22487732
14	4	2001-01-14	2.571453959	2.48995735

Sampled value



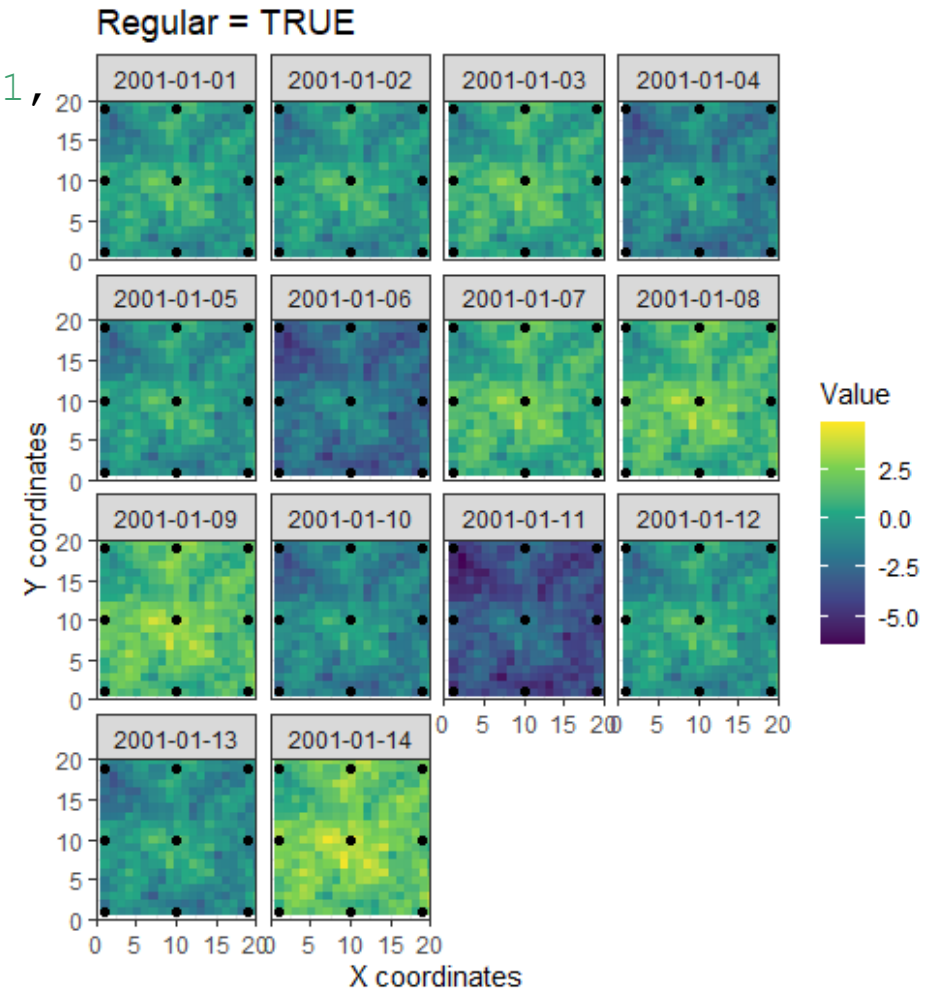
Point sampling

```
#balanced sampling
point_sample <- sample_area_pointily(field= truth,
                                     n_samples=10,
                                     sampling_error=0.1,
                                     regular = FALSE,
                                     balanced = TRUE)
```



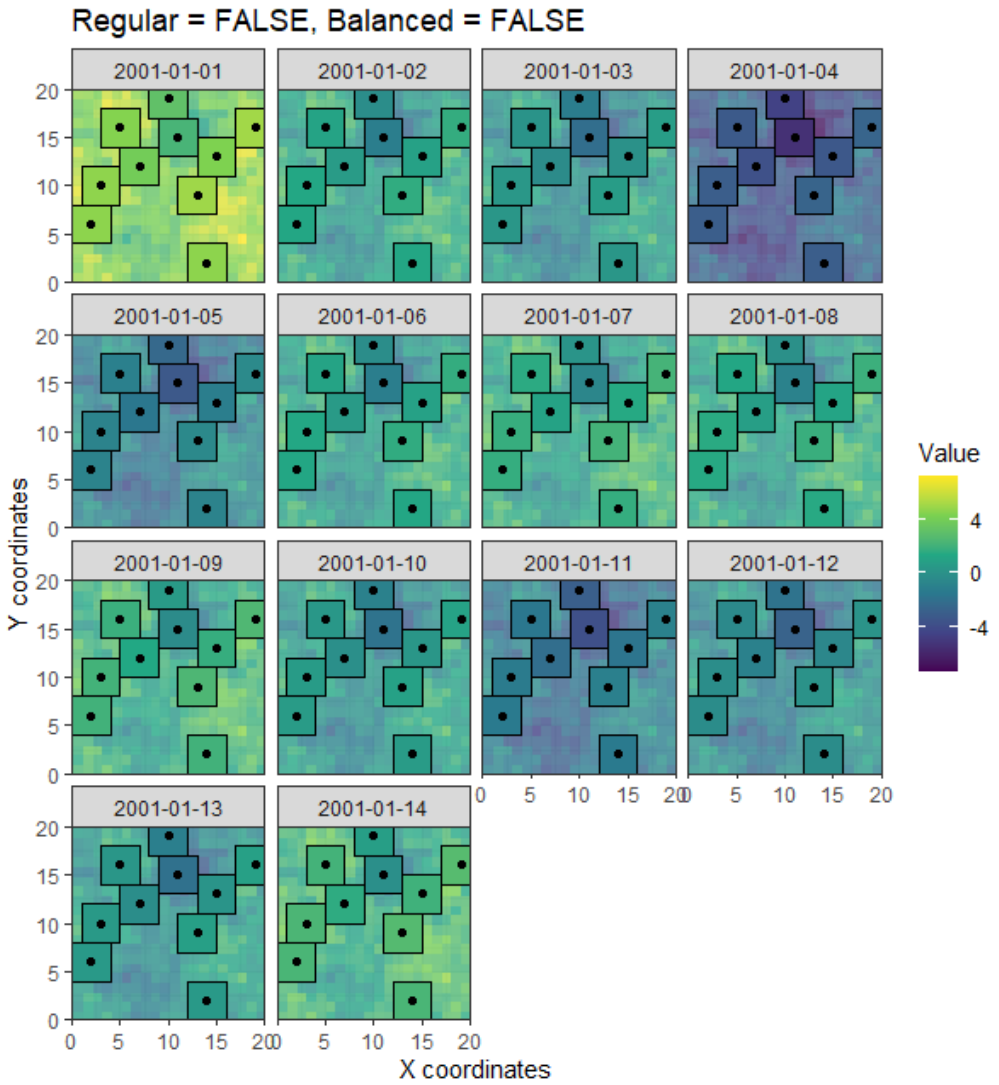
Point sampling

```
#random sampling
point_sample <- sample_area_pointily(field= truth,
                                     n_samples=10,
                                     sampling_error=0.1,
                                     regular = TRUE)
```



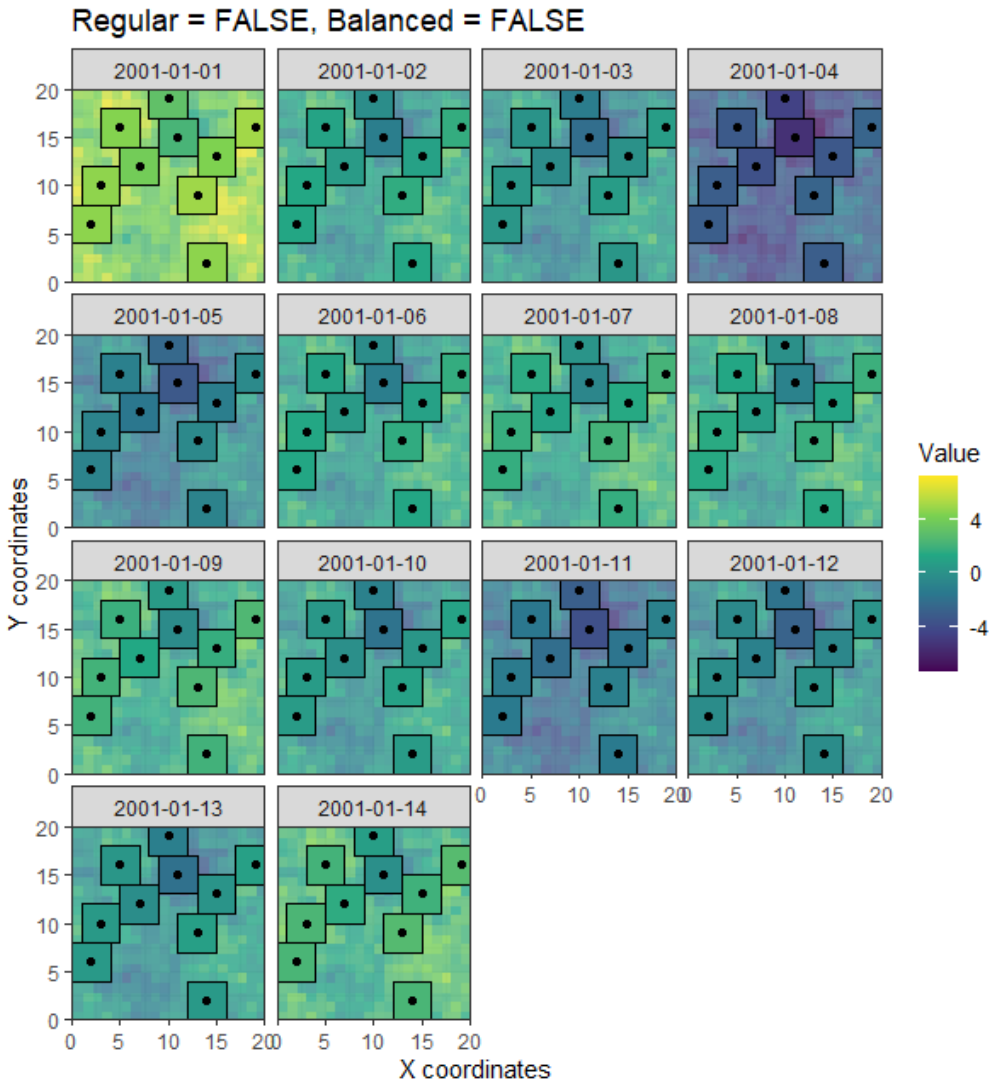
Areal sampling

```
#random areal sampling
area_sample <- sample_area_areally(field=truth,
                                   n_samples =10,
                                   x_range =  2,
                                   y_range = 2,
                                   sampling_error = 0.1,
                                   regular = FALSE,
                                   balanced = FALSE)
```



Areal sampling

```
#random areal sampling
area_sample <- sample_area_areally(field=truth,
                                   n_samples =10,
                                   x_range =  2,
                                   y_range = 2,
                                   sampling_error = 0.1,
                                   regular = FALSE,
                                   balanced = FALSE)
```

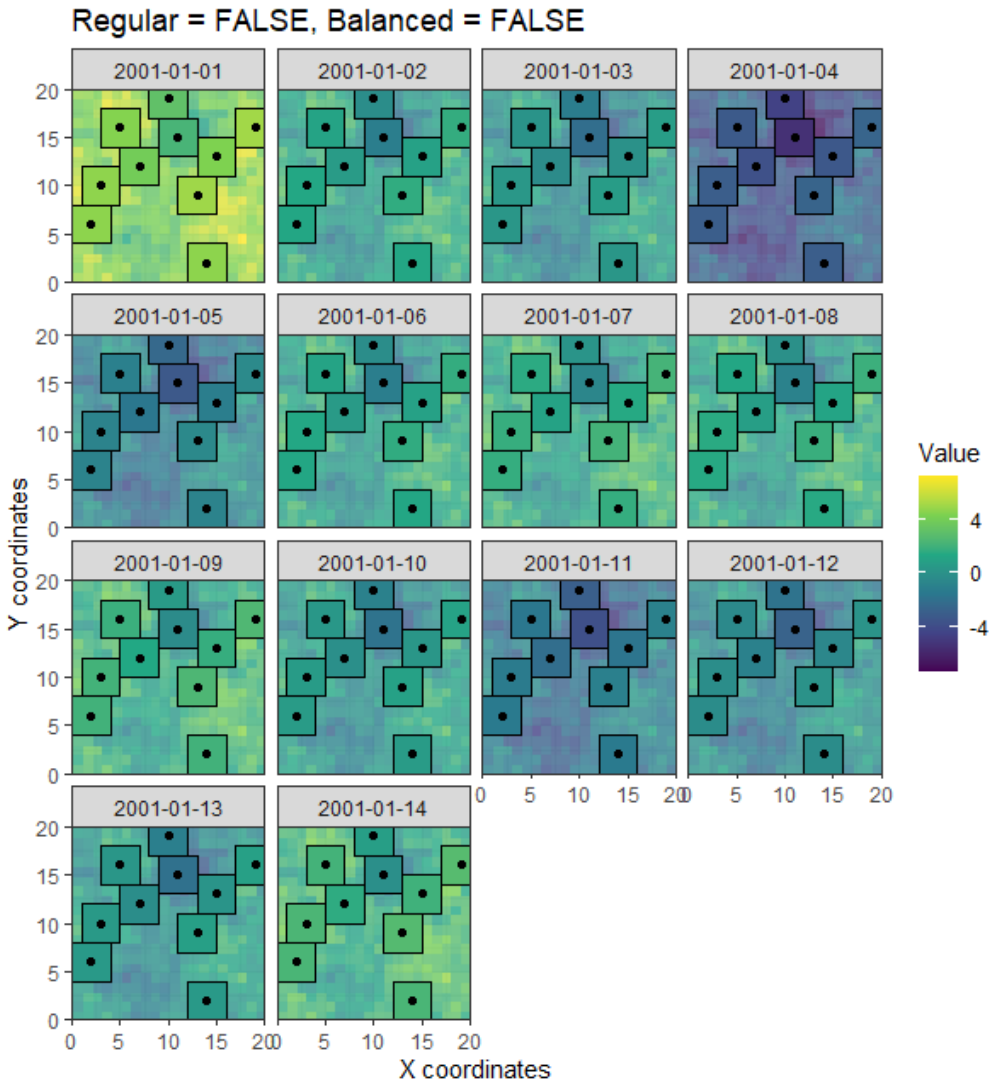


****To avoid overlap between areas, the centroids are resampled each time areas overlap. Please keep this in mind when selecting the number of points to sample and the area size in relation to the total area.****

Areal sampling

```
#random areal sampling
area_sample <- sample_area_areally(field=truth,
                                   n_samples =10,
                                   x_range =  2,
                                   y_range = 2,
                                   sampling_error = 0.1,
                                   regular = FALSE,
                                   balanced = FALSE)
```

date	value	area_value	x	y
2001-01-01	4.86698815	4.985385488	13	9
2001-01-02	1.49821911	1.616616449	13	9
2001-01-03	0.54273091	0.661128246	13	9
2001-01-04	-2.80381413	-2.685416791	13	9
2001-01-05	-0.64119529	-0.522797952	13	9
2001-01-06	1.48914853	1.607545862	13	9
2001-01-07	1.95118591	2.069583243	13	9
2001-01-08	1.68306016	1.801457492	13	9
2001-01-09	2.26730367	2.385701006	13	9
2001-01-10	0.79669139	0.915088723	13	9
2001-01-11	-1.16901040	-1.050613063	13	9
2001-01-12	-0.08256829	0.035829040	13	9
2001-01-13	0.68703443	0.805431765	13	9
2001-01-14	2.51936005	2.637757387	13	9

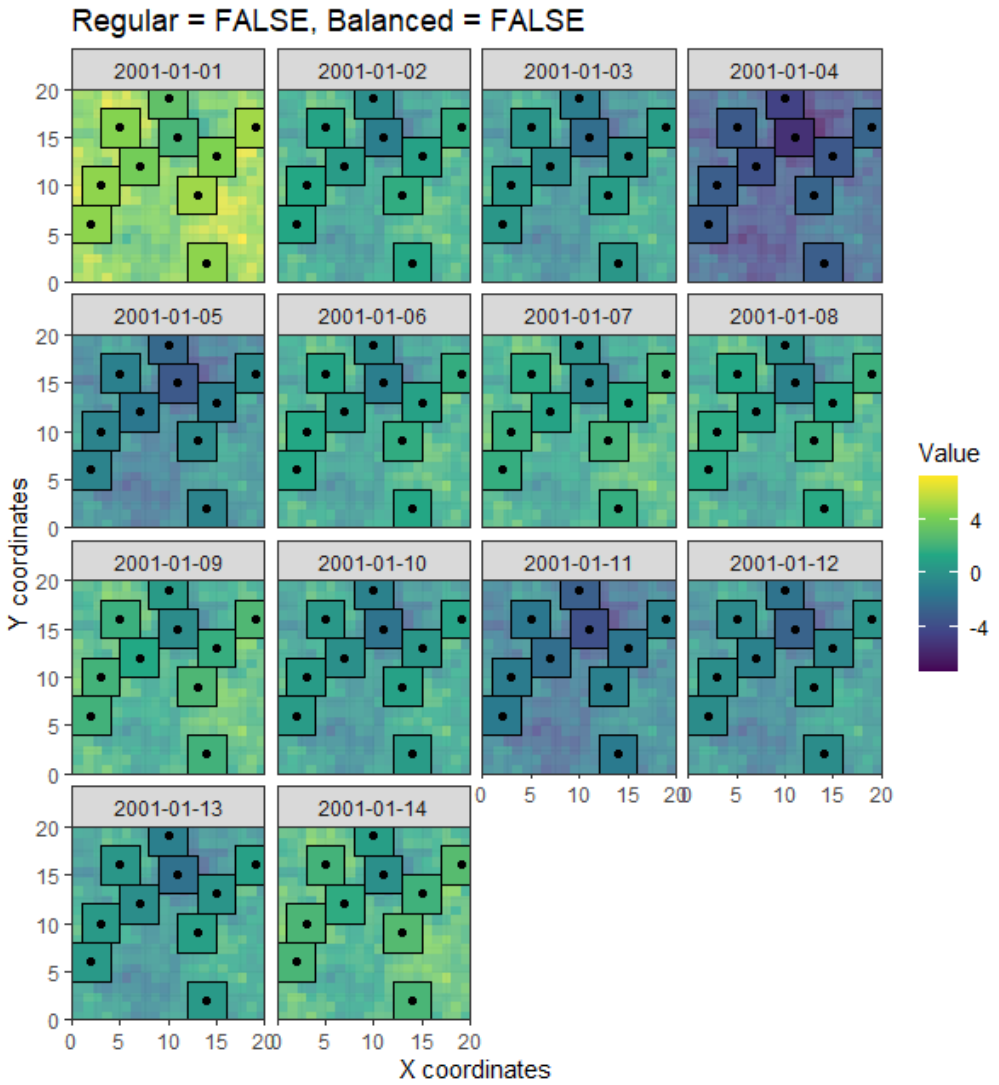


Areal sampling

```
#random areal sampling
area_sample <- sample_area_areally(field=truth,
                                   n_samples =10,
                                   x_range =  2,
                                   y_range = 2,
                                   sampling_error = 0.1,
                                   regular = FALSE,
                                   balanced = FALSE)
```

date	value	area_value	x	y
2001-01-01	4.86698815	[illegible]	13	9
2001-01-02	1.49821911	1.616616449	13	9
2001-01-03	0.54273091	0.661128246	13	9
2001-01-04	-2.80381413	-2.685416791	13	9
2001-01-05	-0.64119529	-0.522797952	13	9
2001-01-06	1.48914853	1.607545862	13	9
2001-01-07	1.95118591	2.069583243	13	9
2001-01-08	1.68306016	1.801457492	13	9
2001-01-09	2.26730367	2.385701006	13	9
2001-01-10	0.79669139	0.915088723	13	9
2001-01-11	-1.16901040	-1.050613063	13	9
2001-01-12	-0.08256829	0.035829040	13	9
2001-01-13	0.68703443	0.805431765	13	9
2001-01-14	2.51936005	2.637757387	13	9

"True" value
(mean of the area)

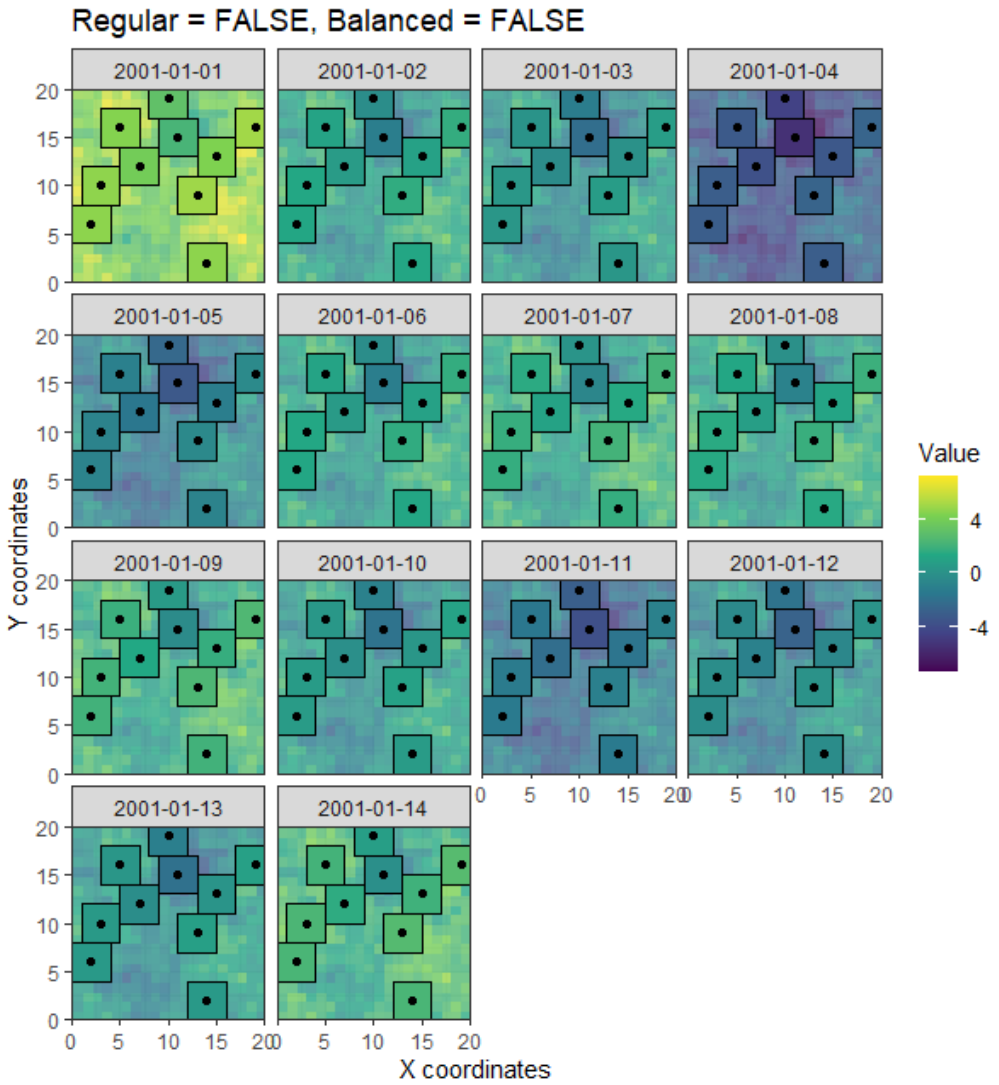


Areal sampling

```
#random areal sampling
area_sample <- sample_area_areally(field=truth,
                                   n_samples =10,
                                   x_range =  2,
                                   y_range = 2,
                                   sampling_error = 0.1,
                                   regular = FALSE,
                                   balanced = FALSE)
```

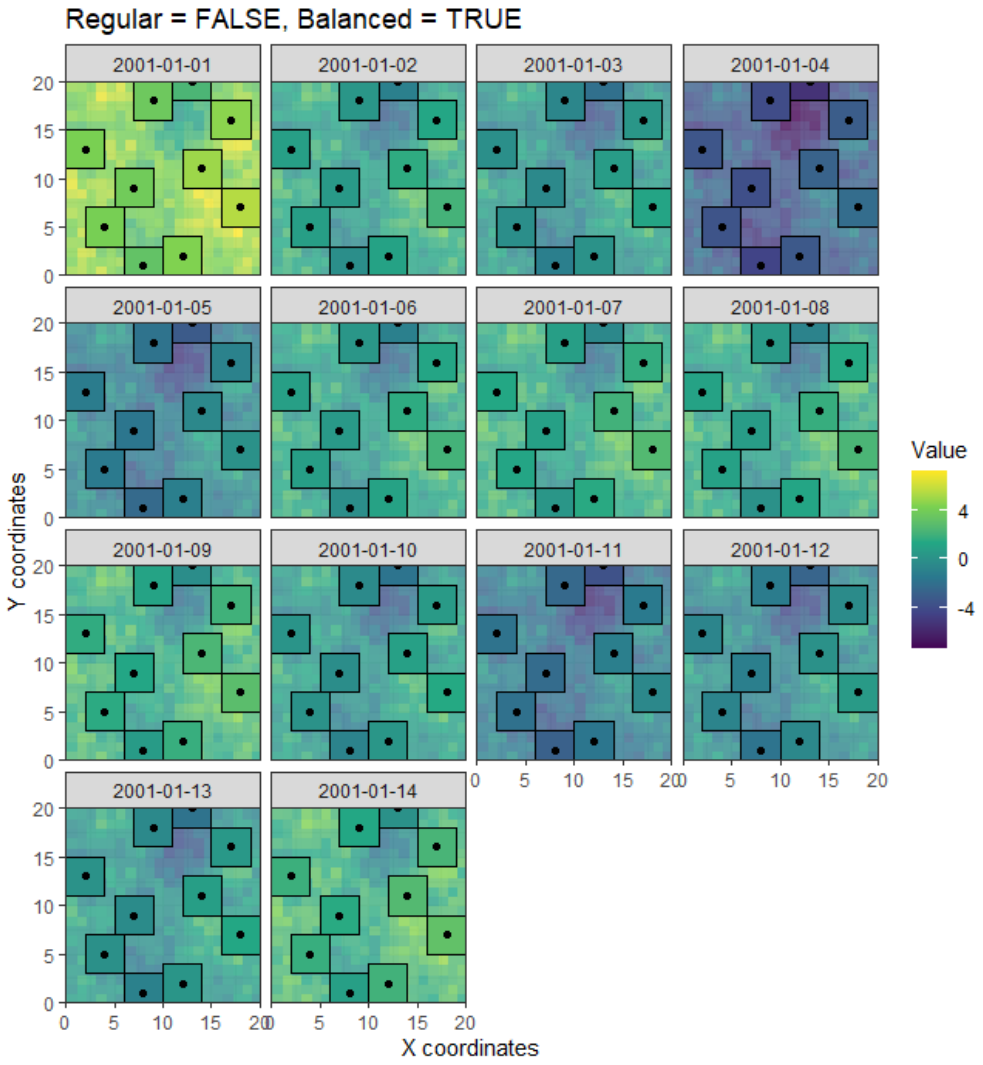
date	value	area_value	x	y
2001-01-01	4.86698815	4.985385488	13	9
2001-01-02	1.49821911	1.616616449	13	9
2001-01-03	0.54273091	0.661128246	13	9
2001-01-04	-2.80381413	-2.685416791	13	9
2001-01-05	-0.64119529	-0.522797952	13	9
2001-01-06	1.48914853	1.607545862	13	9
2001-01-07	1.95118591	2.069583243	13	9
2001-01-08	1.68306016	1.801457492	13	9
2001-01-09	2.26730367	2.385701006	13	9
2001-01-10	0.79669139	0.915088723	13	9
2001-01-11	-1.16901040	-1.050613063	13	9
2001-01-12	-0.08256829	0.035829040	13	9
2001-01-13	0.68703443	0.805431765	13	9
2001-01-14	2.51936005	2.637757387	13	9

Sampled value



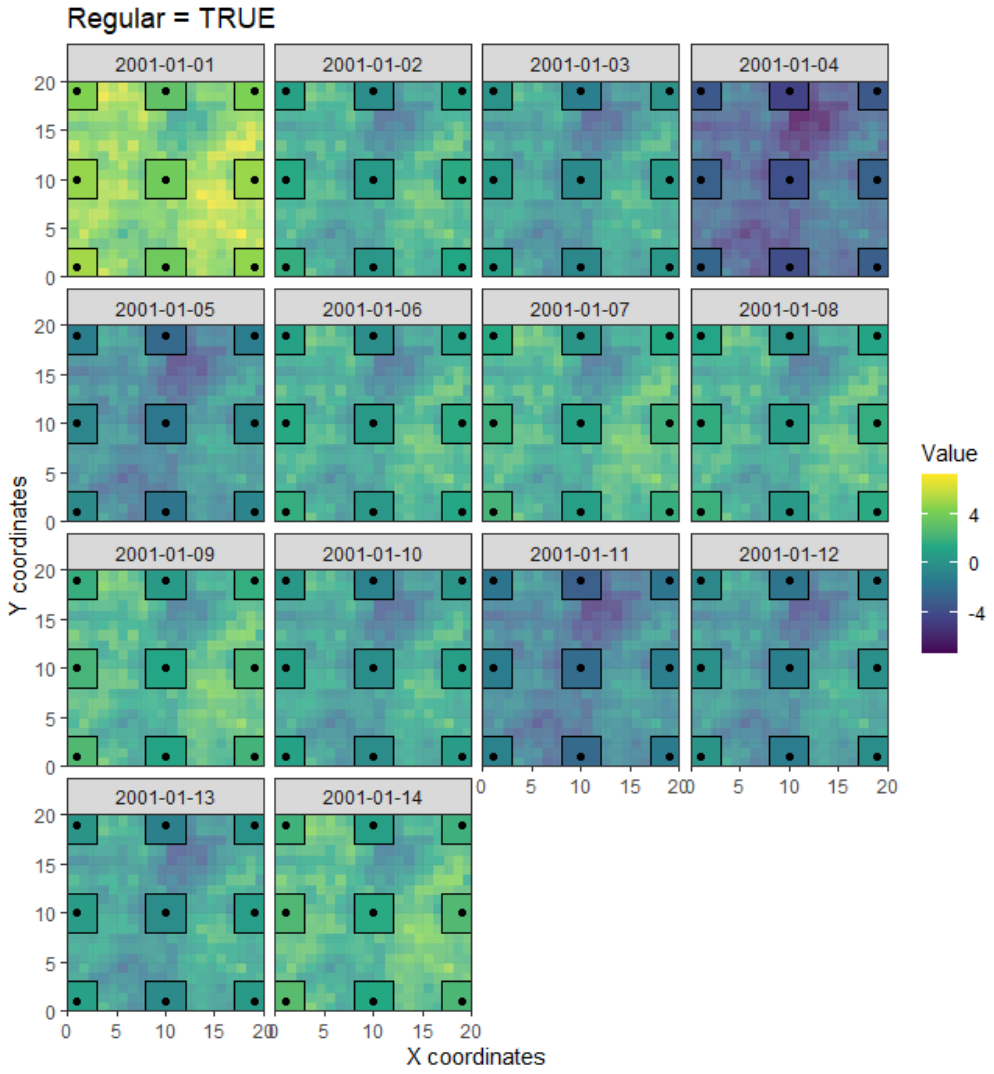
Areal sampling

```
#balanced areal sampling
area_sample <- sample_area_areally(field=truth,
                                   n_samples = 10,
                                   x_range =  2,
                                   y_range = 2,
                                   sampling_error = 0.1,
                                   regular = FALSE,
                                   balanced = TRUE)
```

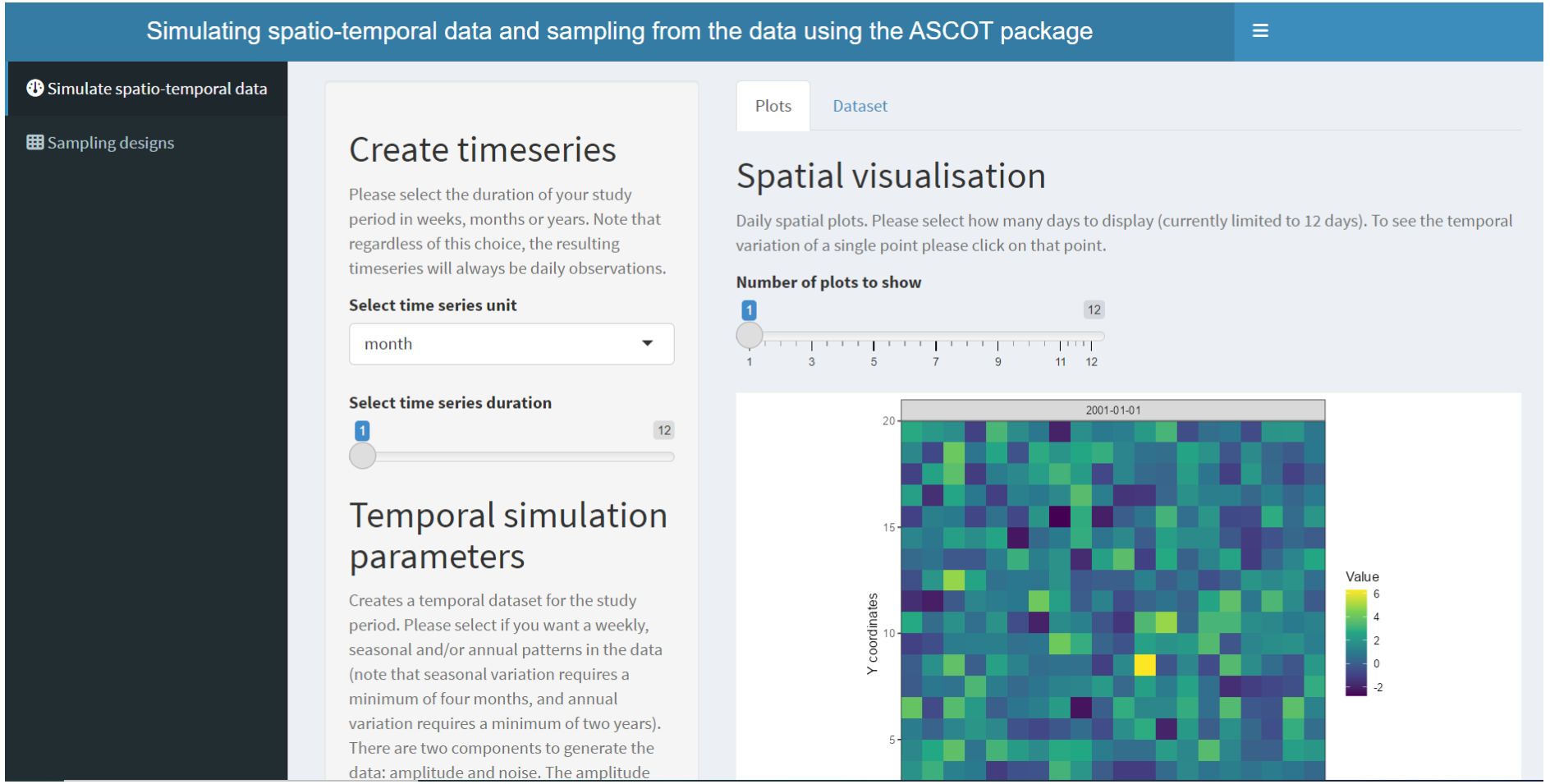


Areal sampling

```
#regular areal sampling
area_sample <- sample_area_areally(field=truth,
                                   n_samples =10,
                                   x_range =  2,
                                   y_range = 2,
                                   sampling_error = 0.1,
                                   regular = TRUE)
```



Shiny app


Simulating spatio-temporal data and sampling from the data using the ASCOT package
Simulate spatio-temporal data
Sampling designs
Create timeseries
Please select the duration of your study period in weeks, months or years. Note that regardless of this choice, the resulting timeseries will always be daily observations.
Select time series unit
month
Select time series duration
1
12
Temporal simulation parameters
Creates a temporal dataset for the study period. Please select if you want a weekly, seasonal and/or annual patterns in the data (note that seasonal variation requires a minimum of four months, and annual variation requires a minimum of two years). There are two components to generate the data: amplitude and noise. The amplitude
Plots
Dataset
Spatial visualisation
Daily spatial plots. Please select how many days to display (currently limited to 12 days). To see the temporal variation of a single point please click on that point.
Number of plots to show
1
12
1
3
5
7
9
11
12
2001-01-01
Y coordinates
20
15
10
5
Value
6
4
2
0
-2

Question?