ESSnet Big Data II

Grant Agreement Number: 847375-2018-NL-BIGDATA

https://webgate.ec.europa.eu/fpfis/mwikis/essnetbigdata
https://ec.europa.eu/eurostat/cros/content/essnetbigdata_en

Workpackage I
Mobile Network Data

Deliverable I.8
Visualisation tools for the production of official statistics with mobile network data

Final version, 18 March, 2021

Prepared by:

Martijn Tennekes (CBS, The Netherlands)

- Sandra Barragán (INE, Spain)
- Bogdan Oancea (INS, Romania)
- David Salgado (INE, Spain)
- Luis Sanguiao (INE, Spain)

Workpackage Leader:

David Salgado (INE, Spain)
david.salgado.fernandez@ine.es
telephone : +34 91 5813151
mobile phone : N/A

Contents

1

Introduction

In this document we provide a description of the visualisation tools associated to the methodological framework built within the work package I of the ESSnet Big Data II project with the purpose of incorporating mobile network data into the production of official statistics. This document contains no explanation whatsoever about the underlying statistical methodology. Thus, an interested reader should be familiarized with the content of the proposed methodological approach which is given in Deliverable I.3 (Methodology) - A proposed production framework with mobile network data (Salgado et al., 2020), in the technical document by Tennekes et al. (2020), and the documentation of package `mobloc` (Tennekes, 2020a). However, we shall provide a brief description of the sort of data we want to visualise and of the step of the end-to-end production process we shall be focusing on with these tools.

The document is divided into the following chapters. In this short introductory chapter we just present an overall motivation of the software implementation for visualization in agreement with deliverable I.4 on IT (Oancea et al., 2020) together with a brief introduction to the mobile network data and the process step we will concentrate on. Although developing a common framework for visualization, we have divided this document into two main chapters with a view on research and development, on the one hand, and on production, on the other hand. Thus, chapter 3 shows how synthetic network event data generated by the simulator in this ESSnet (Oancea et al., 2019) can be easily visualised. In parallel, chapter 2 shows how to use this framework to visualise data in a production context, with more complete and realistic data sets. Finally, in chapter 4 we comment on future prospects.

Regarding the selection of the programming language for our software implementation the approach is completely aligned with the software development conducted in deliverable I.4 (see Oancea et al., 2020):

- Portability.
- Software for statisticians, not for computer scientists.
- Open source software.
- Language with support for parallel and distributed computing.
- Programming efficiency and resources.

Thus, R is the selected language so that we have developed a standalone R package called `mobvis` (Tennekes, 2020b).

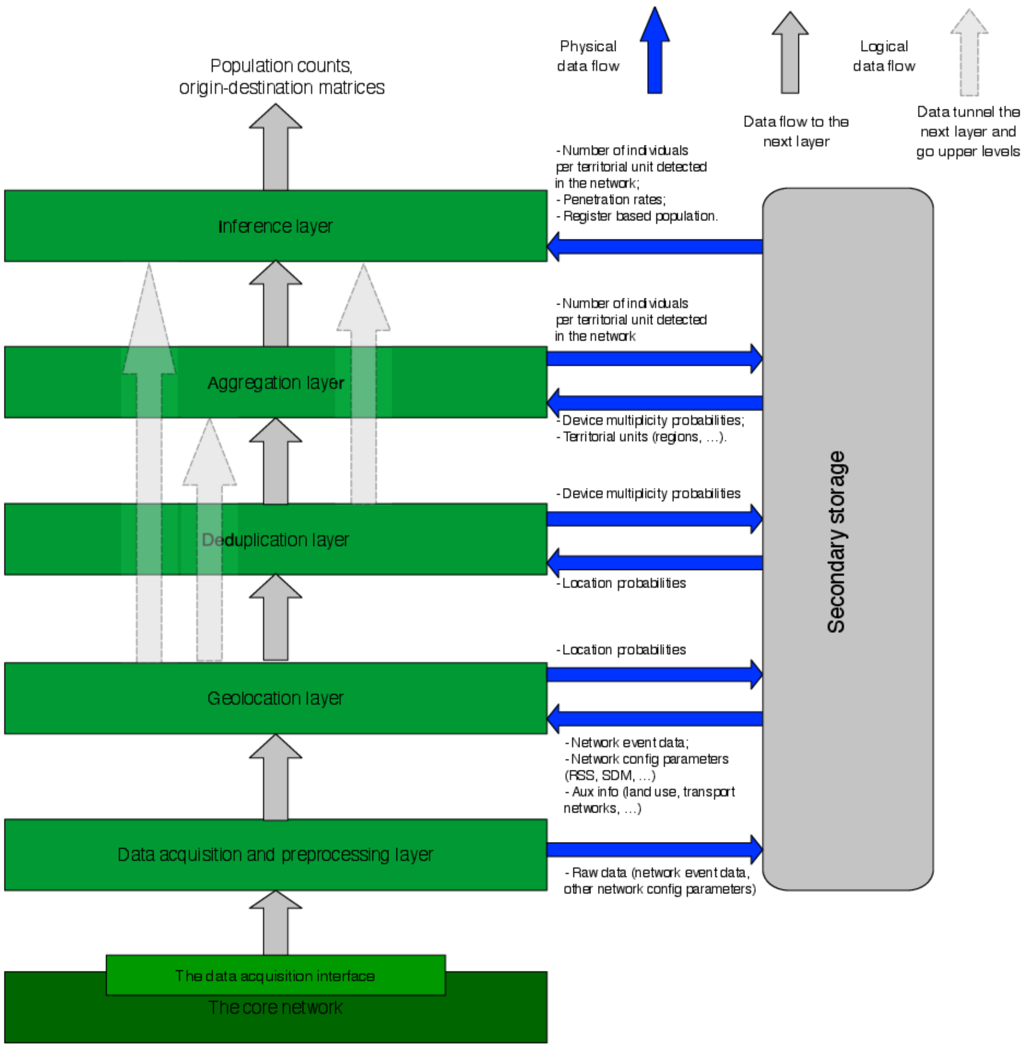

Figure 1.1 End-to-end modular statistical process from raw telco data to final estimates (present population counts and origin-destination matrices in our case).

1.1. Mobile network data: geolocation

Mobile network data is a generic term to refer to a varied sort of data generated in a mobile telecommunication network. We distinguish the following sets (see Radini et al., 2020):

- network event data, for data generated by the interaction between each mobile device and the Base Transceiver Stations (BTSs) of the network;
- network configuration data (sometimes also referred to as cell plan), for data expressing the configuration of the network such as BTS position coordinates, emission power, loss exponent, etc.;
- telecommunication market data, for data about penetration rates and market shares at diverse degree of spatial and time breakdown, etc.;
- official statistical data, for data (both at an aggregate and microdata degrees of breakdown) produced by a statistical office such as a population register, etc.;
- public data, for data produced by any other public institution such as the transport network or weather conditions.

For research and development, we must also consider data for the synthetic ground truth produced by the network event data simulator (Oancea et al., 2019). Apart, the end-to-end production process generates intermediate data sets which serve as natural interfaces between the modules (see figure 1.1).

In this production environment, being very rich in data, there arise many options to develop visualization tools. We have focused on the geolocation module, especially on visualising the location probabilities computed for any given device along its movement across the geographical territorial. When working with the network event data simulator, besides, we shall also visualise the true movement and position, so that we can visually appraise the quality of the estimation method for the location. Additionally, both the network configuration and the subsequent event location probabilities will also be visualised with these tools.

2

Visualising mobloc data

The **mobloc** package is used to approximate geographic location given the connection between a mobile device and the cell (an antenna may contain multiple cells). This is done by a physical model of signal strength and a Bayesian method to estimate the location of mobile devices. The latter is done by applying Bayes' formula

$$\mathbb{P}(g|a) \propto \mathbb{P}(g)\mathbb{P}(a|g)$$

where

- $\mathbb{P}(g|a)$ is a condensed notation for the *posterior location probability* $\mathbb{P}\left(T_d = g \middle| \mathbf{E}_d = a, \mathbf{I}^{\text{aux}}\right)$ that a device d is located in grid tile $T_d = g$ given that it is connected to cell $\mathbf{E} = a$ (see Salgado et al., 2020, for the detailed notation);
- $\mathbb{P}(g)$ is a condensed notation for the *prior* probability $\mathbb{P}\left(T_d = g \middle| \mathbf{I}^{\text{aux}}\right)$ that a device d is located in grid tile $T_d = g$. The most simple model is to set it to a constant value. Alternatively, external data sources such as land use can be used;
- $\mathbb{P}(a|g)$ is a condensed notation for the *likelihood* (event location probability) $\mathbb{P}\left(\mathbf{E}_d \middle| T_d = g, \mathbf{I}^{\text{aux}}\right)$, which is the probability that a device d is connected to cell $\mathbf{E}_d = a$ given that is it located in grid tile $T_d = g$. For the computation of these probabilities, mobloc offers two methods: the first uses Voronoi, which assumes that a device will always be connected to the nearest cell, and the second uses the signal strength model. In a wider context with dynamical considerations (Salgado et al., 2020), these probabilities are also called *event location probabilities*.

These methods are described in detail by Tennekes et al. (2020). The implementation is described in the documentation of the **mobloc** package (Tennekes, 2020a).

2.1. Processing example data in mobloc

The **mobloc** package contains fictional **cellplan** dataset called `ZL_cellplan`. A cellplan is a dataset that contains the locations and physical properties of the cells from a certain MNO. The cells in `ZL_cellplan` are fictionally placed in the region of Zuid-Limburg, which is NUTS region NL423. Besides the cellplan, other datasets that are contained in **mobloc** are the municipality polygons for Zuid-Limburg `ZL_muni`, the elevation data `ZL_elevation`, and land use data `ZL_landuse`.

These datasets area loaded as follows.

```
library(mobloc)
data("ZL_cellplan", "ZL_muni", "ZL_elevation", "ZL_landuse")
```

ZL_cellplan and ZL_muni are **sf** (Pebesma, 2018) objects of points and polygons respectively. The objects ZL_elevation and ZL_landuse are **raster** (Hijmans, 2020) objects.

The following code processes these input datasets.

```
# set parameters
ZL_param <- mobloc_param()

# create environment layer (needed to calculate path loss exponent (ple))
ZL_envir <- combine_raster_layers(ZL_landuse, weights = c(1, 1, 1, 0, 0))

# validate cellplan
ZL_cellplan <- validate_cellplan(ZL_cellplan, param = ZL_param,
    region = ZL_muni,envir = ZL_envir, elevation = ZL_elevation)
#> Variable 'W' is missing. These have been imputed with the parameters 'W'
#> and 'W_small' for normal and small cells respectively.
#> Variable 'range' is missing. These have been imputed with the parameters
#> 'range' and 'range_small' for normal and small cells respectively.
#> Path loss exponent ('ple') values updated with envir object
#> The cellplan has been made valid

# create raster
ZL_bbox <- sf::st_bbox(
   c(xmin = 4012000, ymin = 3077000, xmax = 4048000, ymax = 3117000),
   crs = sf::st_crs(3035))
ZL_raster <- create_raster(ZL_bbox)

# compute the signal strength model
ZL_strength <- compute_sig_strength(cp = ZL_cellplan, raster = ZL_raster,
    elevation = ZL_elevation, param = ZL_param)
#> Determining coverage area per cell
#> Determine signal strength per cell for raster tiles inside coverage area
#> Creating data.frame and compute pag values

# create likelihoods (event locations)
ZL_strength_llh <- create_strength_llh(ZL_strength, param = ZL_param)
ZL_voronoi_llh <- create_voronoi_llh(ZL_cellplan, ZL_raster)

# create priors
ZL_uniform_prior <- create_uniform_prior(ZL_raster)
ZL_network_prior <- create_network_prior(ZL_strength, ZL_raster)
ZL_landuse_prior <- create_prior(ZL_landuse, weights = c(1, 1, .1, 0, .5))

# create composite prior: half network, half lansuse
ZL_composite_prior <- create_prior(ZL_network_prior,
    ZL_landuse_prior, weights = c(0.5, 0.5))

# caculate the posterior distributions
ZL_posterior <- calculate_posterior(prior = ZL_composite_prior,
    llh = ZL_strength_llh, raster = ZL_raster)
```

This code is explained in the documentation of the **mobloc** package. The most important objects are:

- `ZL_param`, a list that contains all the parameters used in **mobloc**, e.g. the default physical properties of the cells.
- `ZL_raster`, a `raster` object that contains the geospatial information of the grid tiles, e.g. their spatial position and the resolution.
- `ZL_strength`, a `data.table` object that contains the signal strength in `dBm` and the signal dominance (called `s`) per grid tile.
- `ZL_xxx_llh`, `data.table` objects that contain the likelihood probabilities (event locations) per cell. This probability is called `pag`, which stands for `p(a|g)`, which is the probability that a device is connected to cell `a`, given that it is located in grid tile `g`.
- `ZL_xxx_prior`, `raster` objects in the same resolution as `ZL_raster`, that contain prior information. This probability is called `pg`, which stands for `p(g)`, the probability that a device is located in grid tile `g`.
- `ZL_posterior`, a `data.table` that contains the posterior probabilities. This probability is called `pga`, which stands for `p(g|a)`, which is the probability that a device is located in grid tile `g`, given that it is connected to cell `a`.

2.2. Visualization of mobloc parameters

```
library(mobvis)
```

The following function starts a dashboard that examines the parameters used in mobloc.

```
setup_sig_strength_model()
```

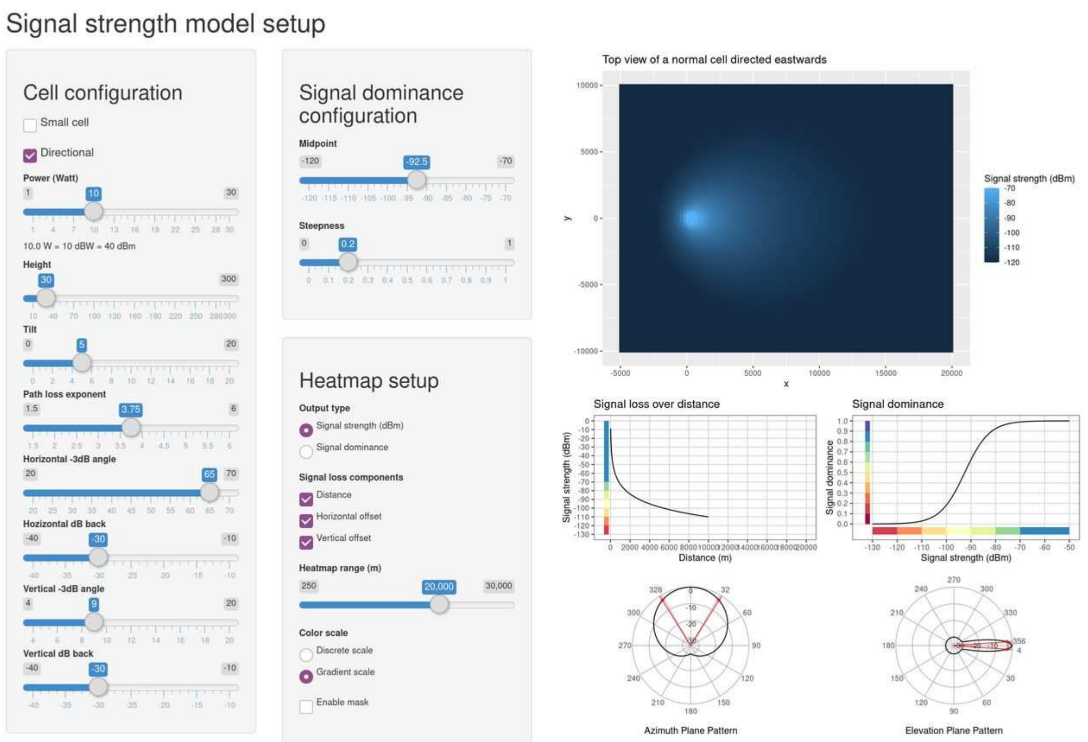

Figure 2.1 Setup panel for cell configuration parameters.

This tool (see figure 2.1) shows the signal strength model for one cell.

The left hand side panel in figure 2.1 shows the settings, which are by default set to values of the list we created with `mobloc_param()`. The plots on the right hand side shows the propagation results. The heatmap on the top right shows the top view of the signal strength of the cell. The direction of directional cells is east in this plot. The four plots below the heatmap can be reproduced with the following single function calls.

The function `distance_plot` plots the relation between signal strength (in dBm) and distance in the propagation direction of the cell. This depends on the cell power in watt (`W`) and the path loss exponent (`ple`) which models how well the signal propagates (2 is free space, 4 is urban area, and is inside buildings).

```
distance_plot(W = 4, ple = 4, range = 2000)
```

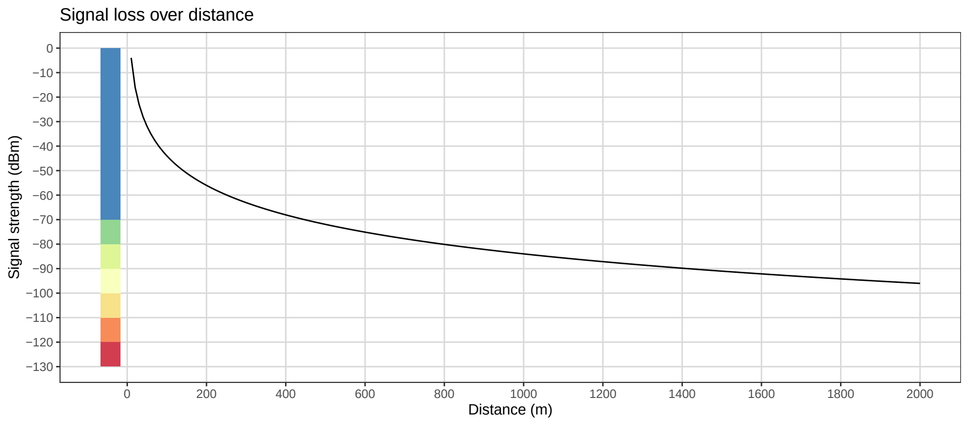

Figure 2.2 Signal strength (in dBm) vs. distance to the cell.

The color codes on the y-axis resemble how well the signal is for mobile communication. This will be further discussed later on.

The function `signal_dominance_plot` plots the modeled relation between signal strength (in dBm) and the signal dominance (s).

```
signal_dominance_plot(midpoint = -90, steepness = 0.5)
```

Signal dominance can be interpreted as the quality of a connection between device and mobile phone, or in other words, how attractive it is to make create connection. It is based on the assumption that for a mobile phone network, it is not important whether the connection is good (say -90 dBm) or extreme good (say -70 dBm). When there are multiple cells to choose from, the current capacity is much more important. Likewise, if there is only one available cell for a device to connect to at a certain location, the network will do that, no matter whether the connection is fair (say -100 dBm) or bad (say -120 dBm). This has been modeled by flattening both tails of the distribution. A logistic function has been applied to do this (Tennekes et al., 2020), where its parameters are midpoint and steepness.

The function `radiation_plot` plots the modeled radiation pattern from the cell (see figure 2.4). The radiation pattern indicates how much signal loss there is in the azimuth plane (top view perpendicular to the propagation direction), and the elevation plane (side view perpendicular

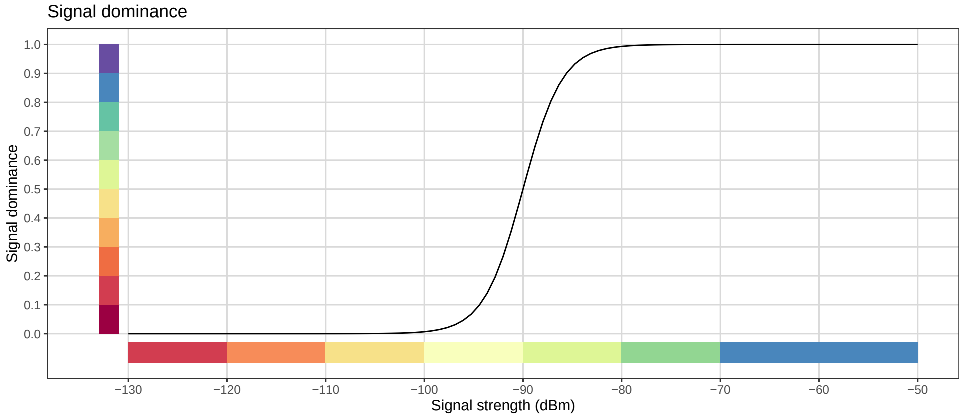

Figure 2.3 Signal dominance measure vs. distance to the cell.

to the propagation direction). The black contour lines indicate the signal loss as a function of the offset angle. The red points are the -3dB points, i.e. the angle at which the signal loss is 3 dB. The three parameters for this pattern are `type` which indicated whether it is applied to the azimuth (`"a"`) or elevation (`"e"`) place, `beam_width`, which specifies the angle at which the signal loss is 3 dB (so the angle between the red points in the diagram), and `db_back` which defines the signal loss the the angle in the opposite direction of the propagation direction.

```
radiation_plot(type = "a", beam_width = 60, db_back = -30)
radiation_plot(type = "e", beam_width = 10, db_back = -30)
```

Figure 2.4 Radiation plots in terms of azimuth (left) and elevation (right).

2.3. Visualization of the mobloc output

The dashboard that examines the processed data from **mobloc** (see figure 2.5) can be called as follows:

```
# explore the results
explore_mobloc(
    cp = ZL_cellplan,
    raster = ZL_raster,
    strength = ZL_strength,
    priorlist = list(landuse = ZL_landuse_prior,
                     network = ZL_network_prior,
                     uniform = ZL_uniform_prior),
    llhlist = list(Strength = ZL_strength_llh, Voronoi = ZL_voronoi_llh),
    param = ZL_param)
```

In the left-hand-side panel, the user is able to select which raster variable is shown on the map. The options are listed on the left-most column. They are: signal strength (dBm), signal dominance (s), best server map, prior, likelihood, and posterior. For each of them there is a stand-alone function which will be shown and described below. The first option, called "Selection" determines whether the data is shown for all cells or one single cell. In the former case, the maximum number per grid tile is shown. For instance, if the signal strength in dBm is selected for all cells, the color of a grid tile represents the maximum signal strength.

The second column, called "Module Setup" configures the prior, the likelihood, the posterior (which uses the chosen prior and likelihood), and whether Timing Advance is used. In the other tab panel, called "Cell data", cellplan data per cell is shown in a table.

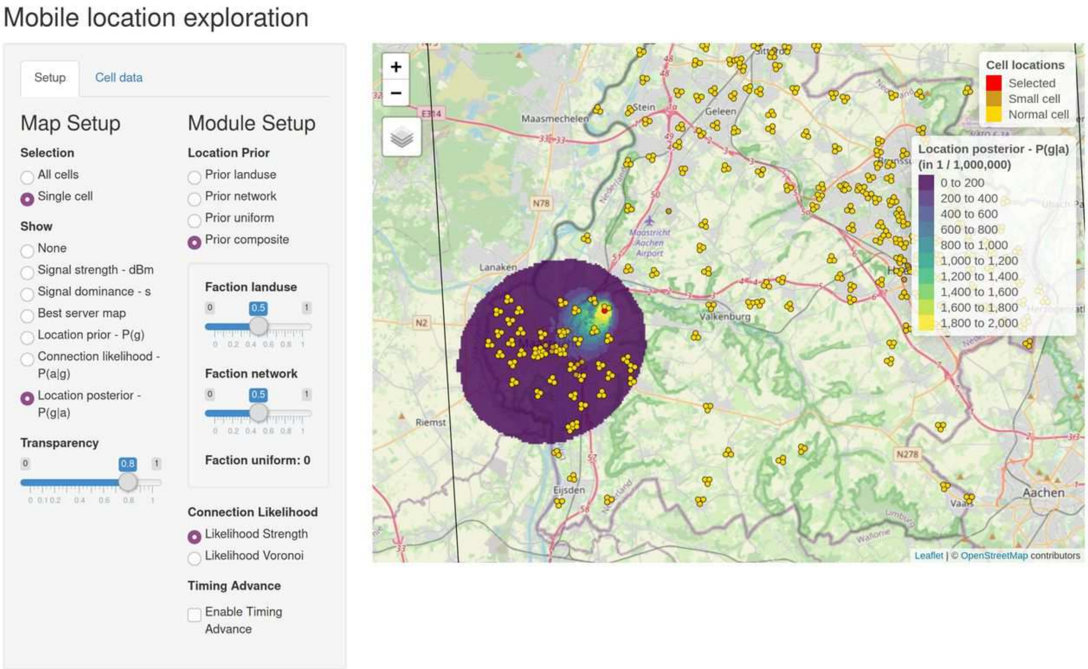

Figure 2.5 Dashboard for output visualization.

2.3.1. Signal strength and signal dominance

The function `map_sig_strength` is used to map the signal strength or dominance. Which of these two will be shown depends on the `type` argument.

```
map_sig_strength(rst = ZL_raster,
                 dt = ZL_strength,
                 cp = ZL_cellplan,
                 cells = "BEE_150_N1",
                 region = ZL_muni,
                 type = "dBm",
                 interactive = FALSE)
map_sig_strength(rst = ZL_raster,
                 dt = ZL_strength,
                 cp = ZL_cellplan,
                 cells = "BEE_150_N1",
                 region = ZL_muni,
                 type = "dBm",
```

```
                  interactive = FALSE,
                  settings = mobvis_settings(use_classes = FALSE))
map_sig_strength(rst = ZL_raster,
                  dt = ZL_strength,
                  cp = ZL_cellplan,
                  cells = "BEE_150_N1",
                  region = ZL_muni,
                  type = "s",
                  interactive = FALSE)
map_sig_strength(rst = ZL_raster,
                  dt = ZL_strength,
                  cp = ZL_cellplan,
                  cells = "BEE_150_N1",
                  region = ZL_muni,
                  type = "s",
                  interactive = FALSE,
                  settings = mobvis_settings(use_classes = FALSE))
```

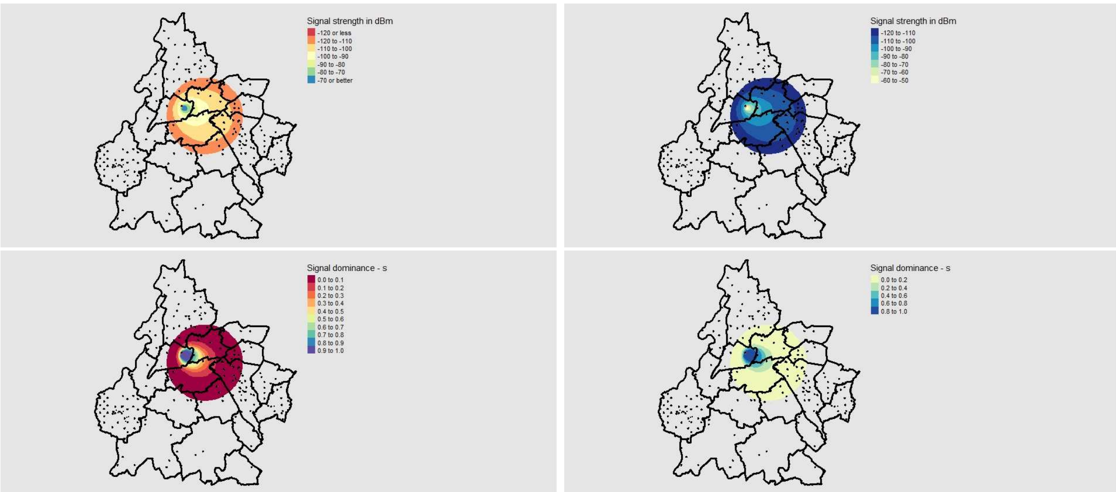

Figure 2.6 Signal strength in dBm (top row) and Signal Dominance Measure (bottom row) with different visualization options (colours).

The function `mobvis_settings()` returns a list that contains settings for the plots drawn by **mobvis**. For instance default color palettes, titles, and whether the plot is static (such as shown here) or interactive (shown in the screenshot of the dashboard tool). The function `mobvis_settings_interactive()` returns the same list, with a few different values, which are aimed for interactive plots. For instance, the cells are by default drawn in black in static maps and yellow with a black border in interactive maps. The function `mobvis_settings_animation()` contains settings for animations, which are described in chapter 3.

As for signal strength and signal dominance, by default a diverging color palette is shown, from red via yellow to blue, which qualifies the signal strength/dominance as more or less as follows: red is bad, orange is poor, yellow is fair, green is good, and blue is excellent. The mappings (so which values are bad, poor, etc.) can be adjusted via `mobvis_settings`. There is also a setting that disables these classes, which is in the maps on the right hand side.

2.3.2. Prior, likelihood and posterior

The prior, likelihood and posterior can be shown with the functions `map_pg`, `map_pag`, and `map_pga` respectively. See figure 2.7.

```
map_pg(rst = ZL_composite_prior,
       cp = ZL_cellplan,
       region = ZL_muni,
       interactive = FALSE)
map_pag(rst = ZL_raster,
        dt = ZL_strength_llh,
        cp = ZL_cellplan,
        cells = "BEE_150_N1",
        region = ZL_muni,
        interactive = FALSE)
map_pga(rst = ZL_raster,
        dt = ZL_posterior,
        cp = ZL_cellplan,
        cells = "BEE_150_N1",
        region = ZL_muni,
        interactive = FALSE)
```

Note that in these functions, the prior is a `raster` object (parameter `rst`) whereas the likelihood and the posterior are `data.tables` (parameter `dt`). The reason of these formats is that the prior distribution is defined on the level of grid tile; recall that the notation is $\mathbb{P}(g)$. The other two, the likelihood (event location) and posterior distribution, are defined per grid tile per cell, in mathematical notation $\mathbb{P}(ajg)$ and $\mathbb{P}(gja)$, respectively.

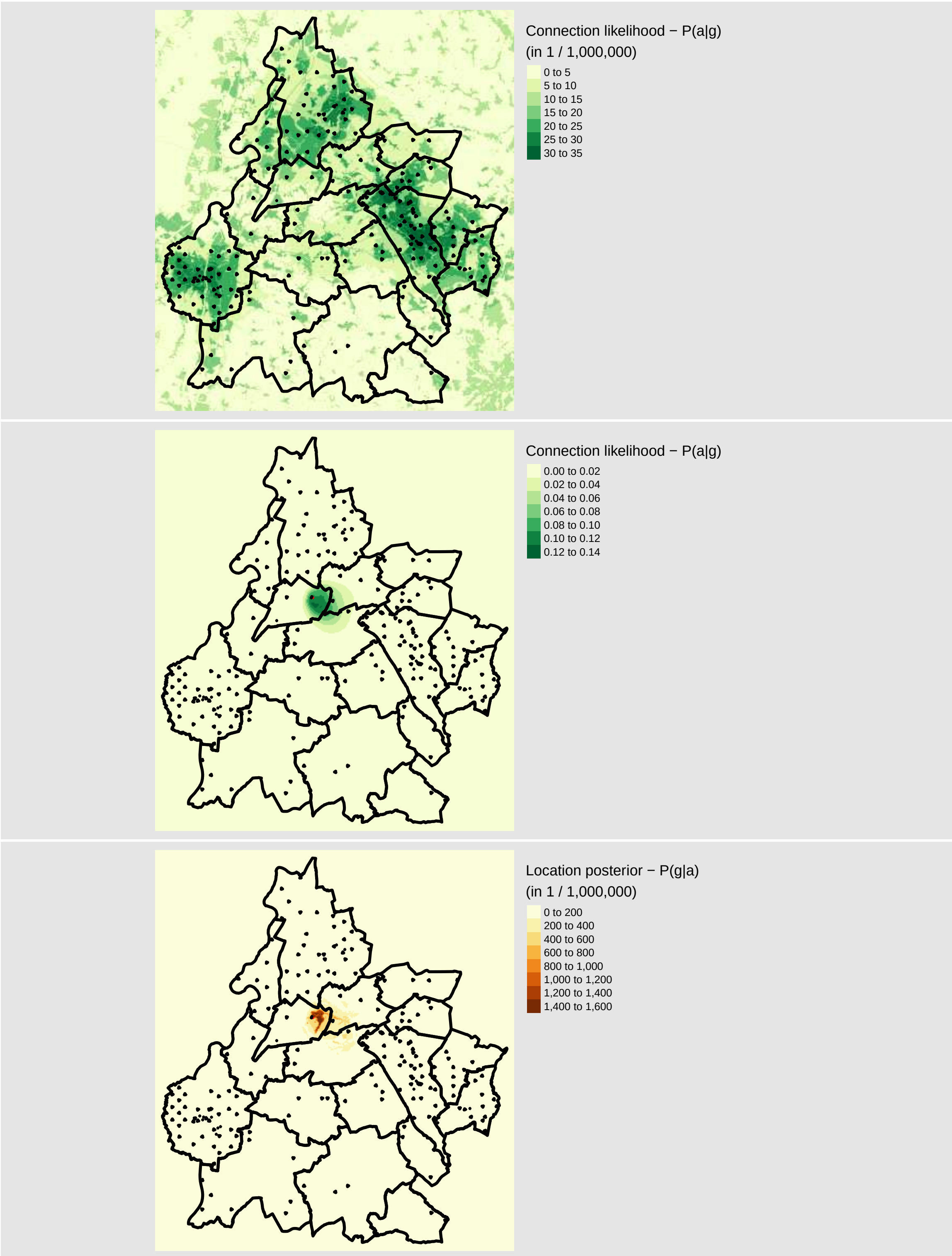

Figure 2.7 Map representations of prior (top), event location (middle), and posterior (bottom) probabilities.

3

Visualising simulator data

The simulator software is a framework to run mobile network data micro-simulations (Oancea et al., 2019). In this vignette, we describe how the results can be plotted with the R package **mobvis**. The package can be loaded in the standard way.

```
library(mobvis)
```

In this chapter, we will also use **tmap** (Tennekes, 2018) to explore spatial objects in general. In the following code chunk, the **tmap** package is loaded, and via a global tmap option, warnings raised by **tmap** are suppress for plotting fictional spatial objects, i.e. that do not have a real geographic location. For this vignette, the github development version is required. This can be installed via `remotes::install_github("mtenneks/tmap")`.

```
library(tmap)
tmap_options(show.warnings = FALSE)
```

3.1. Example data from the simulator

Example data for the simulator, both the input and the output files, are contained in this package as a single zip file. It can be retrieved and extracted as follows.

```
# specify folder for the simulator files
dir <- "simdata"
if (!dir.exists(dir)) {

    dir.create(dir, recursive = TRUE)

} else {

    unlink(dir, recursive = TRUE)
}

# unzip the zip file into that folder
zipfile <- system.file("sim", "simdata.zip", package = "mobvis")
unzip(zipfile, exdir = dir)
```

In this case, the files will be located in the `"simdata"` subfolder of the working directory. The input files are stored in that folder, and the output files in the subfolder `"simdata/output"`.

In order to use this simulator data in **mobvis**, we have to create a list object that contains a few settings:

```
sim <- list(input_dir = "simdata",
            output_dir = "simdata/output",
            mno = "MNO1",
            crs = sf::st_crs(NA))
```

Besides the input and output directories, it also contains the Mobile Network Operator (MNO) name, in this case `"MNO1"`, and a CRS (coordinate reference system). The latter is needed to project the spatial data on the surface of the earth. If the CRS is missing (such as in this example), the coordinate units are still considered to be meters. However, since there is no information about the projection itself, it is not possible to use basemaps in interactive maps.

The following code is used to load the simulator data into R. Note that all the functions in the **mobvis** package that do this have a `sim_get_` prefix.

```
rst <- sim_get_raster(sim)
cp <- sim_get_cellplan(sim)
region <- sim_get_region(sim)
```

The object `rst` is a raster (Hijmans, 2020) object that contains the id numbers of the grid tiles. In correspondence with (Oancea et al., 2019), the numbering goes from left to right and then bottom to top, with 0 being the number for the bottom right tile.

The object `cp` is the cellplan (the dataset that contains the locations and physical properties of the cells from a certain MNO), which is an sf (Pebesma, 2018) object that contains the locations of the cells, and physical properties such as the cell power (in Watt).

The object `region` is an `sf` object that contains the borders of the region of interest.

3.1.1. Visualization parameters

In **mobvis** the visualization settings can be retrieved, and modified via `mobvis_settings()`. It a function that generates a list of visualization settings. An overview of these settings can be seen by looking at the structure with `str()`:

```
str(mobvis_settings())
#> List of 21
#>  $ titles           : Named chr [1:6] "Signal strength in dBm" "Signal dominance - 
#>   ..- attr(*, "names")= chr [1:6] "dBm" "s" "bsm" "pg" ...
#>  $ palettes         :List of 6
#>   ..$ dBm: chr "YlGnBu"
#>   ..$ s  : chr "YlGnBu"
#>   ..$ bsm: chr "Set2"
#>   ..$ pg : chr "YlGnBu"
#>   ..$ pag: chr "YlGn"
#>   ..$ pga: chr "YlOrBr"
#>  $ palette          : chr "YlGnBu"
#>  $ style            : chr "pretty"
#>  $ cell_colors      : Named chr [1:3] "red" "black" "black"
#>   ..- attr(*, "names")= chr [1:3] "Selected" "Small cell" "Normal cell"
#>  $ cell_size        : num 0.5
```

```
#>  $ cell_shape       : num 19
#>  $ cell_offset      : num 75
#>  $ cell_legend      : logi FALSE
#>  $ cell_labels      : logi FALSE
#>  $ cell_label_color: chr "black"
#>  $ region.lwd       : num 3
#>  $ dev_size         : num 0.5
#>  $ dev_color        : chr "purple"
#>  $ dev_shape        : num 15
#>  $ use_classes      : logi TRUE
#>  $ dBm_classes      :List of 5
#>   ..$ breaks: num [1:8] -Inf -120 -110 -100 -90 ...
#>   ..$ labels: chr [1:7] "-120 or less" "-120 to -110" "-110 to -100" "-100 to -90" ...
#>   ..$ colors: chr [1:7] "#D53E4F" "#FC8D59" "#FEE08B" "#FFFFBF" ...
#>   ..$ lims  : num [1:2] -120 -70
#>   ..$ tit   : chr "Signal strength (dBm)"
#>  $ s_classes        :List of 4
#>   ..$ breaks: num [1:11] 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 ...
#>   ..$ labels: chr [1:10] "0.0 to 0.1" "0.1 to 0.2" "0.2 to 0.3" "0.3 to 0.4" ...
#>   ..$ colors: chr [1:10] "#9E0142" "#D53E4F" "#F46D43" "#FDAE61" ...
#>   ..$ lims  : num [1:2] 0 1
#>  $ prob_th          : logi NA
#>  $ bg.color         : chr "grey90"
#>  $ basemaps         : chr "OpenStreetMap"
```

All these settings can easily be changed, as we will demonstrate in the next examples. The visualization settings are described in the package documentation. The default settings are aimed for static maps. There are also default settings for interactive maps and for animations, which can be called via the functions `mobvis_settings_interactive()` and `mobvis_settings_animation()`, respectively.

3.2. Signal strength

The following function reads the signal strength and signal dominance data into R:

```
strength <- sim_get_signal_strength(sim, rst, cp)
print(strength)
#>          cell  rid      dBm            s
#>       1:   70    0 -121.151 8.232145e-17
#>       2:   70    1 -120.824 1.104907e-16
#>       3:   70    2 -120.494 1.487001e-16
#>       4:   70    3 -120.161 2.006640e-16
#>       5:   70    4 -119.826 2.712746e-16
#>     ---
#> 111996:    1 1595 -109.075 6.574383e-08
#> 111997:    1 1596 -109.561 5.156101e-08
#> 111998:    1 1597 -110.032 4.074226e-08
#> 111999:    1 1598 -110.490 3.240348e-08
#> 112000:    1 1599 -110.936 2.592651e-08
```

The columns of `strength` are: cell id number `cell`, tile number `rid`, signal strength `dBm`, and signal dominance `s`.

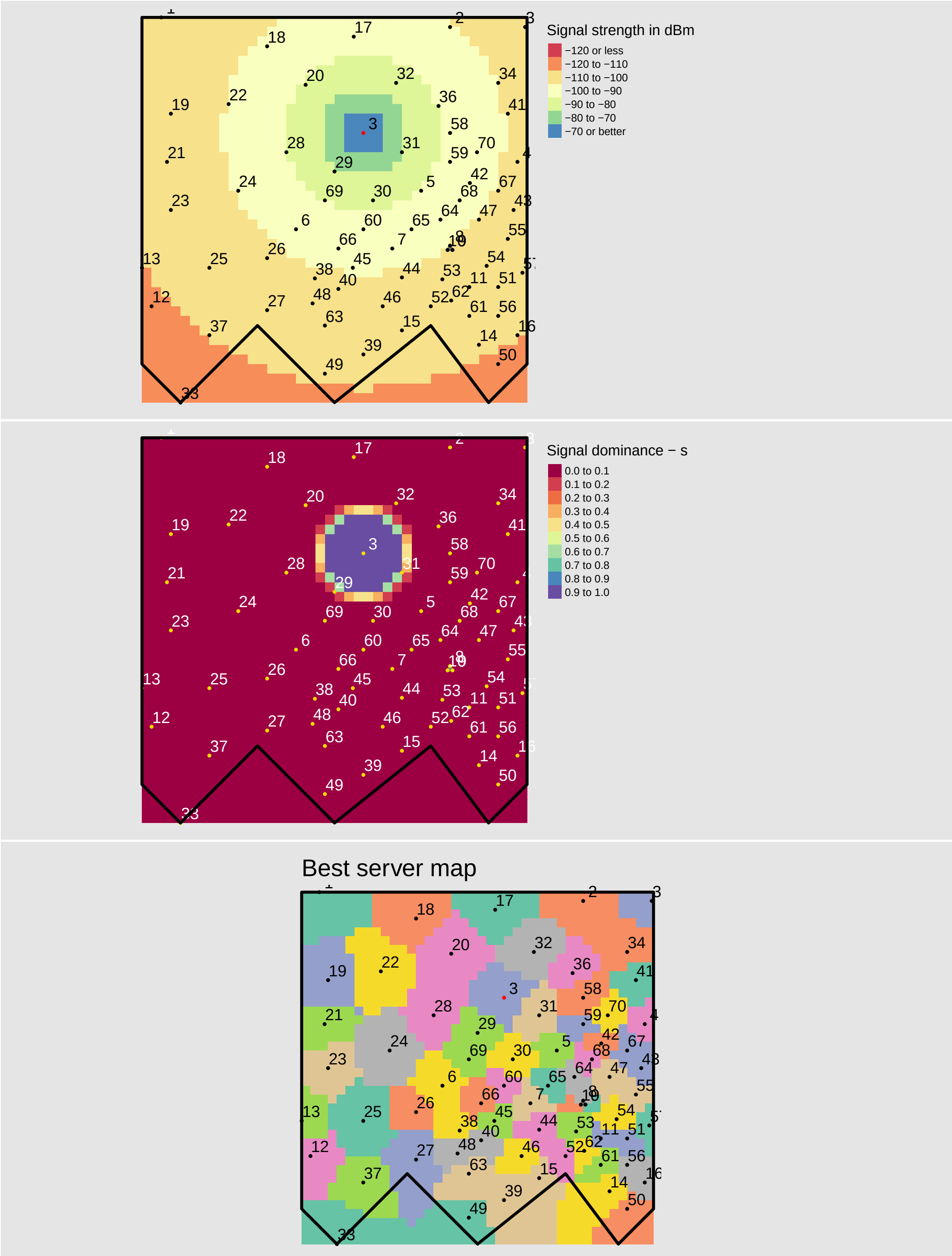

Figure 3.1 Three different map representations of the signal strength in dBm (top), of the Signal Dominance Measure (middle), and of the Best Server Areas (bottom). Each BTS (antenna) is trivially identified with an integer number.

The maps in figure 3.1 are created in the same way as with **mobloc** data as described in chapter 2 using, respectively, the functions `map_pg`, `map_pag`, and `map_pga` (see subsection 2.3.2).

```
map_sig_strength(rst,
                 dt = strength,
                 cp = cp,
                 region = region,
                 cells = 3,
                 type = "dBm",
                 interactive = FALSE,
                 settings = mobvis_settings(cell_size = 1,
                                            cell_labels = TRUE))
map_sig_strength(rst,
                 dt = strength,
                 cp = cp,
                 region = region,
                 cells = 3,
                 type = "s",
                 interactive = FALSE,
                 settings = mobvis_settings(cell_colors = "gold",
                                            cell_size = 1,
                                            cell_labels = TRUE,
                                            cell_label_color = "white"))
map_best_server(rst = rst,
                dt = strength,
                cp = cp,
                region = region,
                cells = 3,
                type = "bsm",
                interactive = FALSE,
                settings = mobvis_settings(cell_size = 1,
                                           cell_labels = TRUE))
```

The last map in figure 3.1, the Best Server Map shows the cell with the best signal strength per tile. Often, this is the nearest cell, but not necessarily.

3.3. Event location

The function `map_pag` visualizes the event location. The event location (likelihood) probabilities are calculated with **mobloc** (see other vignette).

```
library(mobloc)
param <- mobloc_param()
strength[, dist:= NA]
strength_llh <- create_strength_llh(strength, param = param)
print(strength_llh)
#>       cell  rid dist          pag
#>    1:   70  834   NA 0.0008588105
#>    2:   68  834   NA 0.1285867982
#>    3:   67  834   NA 0.1285172165
#>    4:   64  834   NA 0.0878778676
#>    5:   59  834   NA 0.0024428703
#>   ---
#> 8291:    1 1560   NA 1.0000000000
#> 8292:    1 1561   NA 1.0000000000
```

```
#> 8293:     1 1562    NA 1.0000000000
#> 8294:     1 1563    NA 1.0000000000
#> 8295:     1 1564    NA 1.0000000000
```

The **mobloc** requires the signal strength object strength with an additional column `dist`, which stands for distance between tile center and cell location. However, since we do not use `dist` for visualization purposes, we set it to `NA`. The `data.table` object `strength_llh` has four columns cell id number `cell`, tile number `rid`, distance `dist` (not used), and event location (likelihood) probability `pag`.

In the next code chunk, we set the **mobvis** parameters, and create the map in the same way as before.

```
# Retrieve and change the visualization settings
settings = mobvis_settings(cell_size = 2, dev_size = 2)
settings$titles["pag"] <- "Event location"

# Create the plot
map_pag(rst,
        dt = strength_llh,
        cp = cp,
        region = region,
        cells = 3,
        interactive = FALSE,
        settings = settings)
```

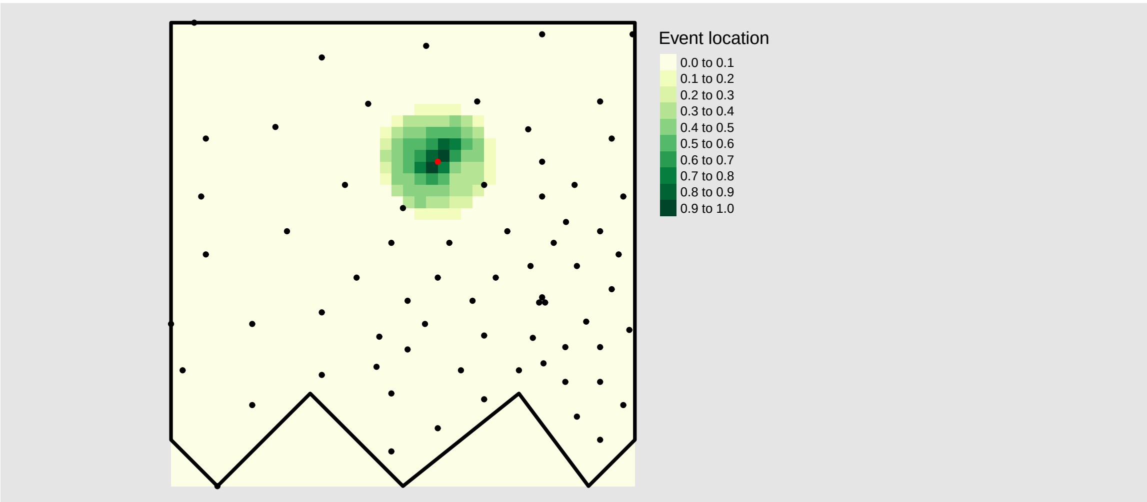

Figure 3.2 Map representation of the event location probabilities. Every black dot represents an omnidirectional BTS (multiple points indicate directional radio cells). The red dot denotes the BTS under visualization (number 3 in the code above).

3.4. Posterior probabilities

The following function visualizes the probability location. The probabilities are also calculated with **mobloc**. The network prior is used.

```
# Calculate the event location (likelihood) probabilities using mobloc
network_prior <- create_network_prior(strength, rst)
posterior <- calculate_posterior(
     prior = network_prior,
     llh = strength_llh,
     raster = rst)
print(posterior)
#>       cell  rid          pga
#>    1:   70  834 0.0001291534
#>    2:   70  835 0.0001168515
#>    3:   70  871 0.0001055452
#>    4:   70  872 0.0004874089
#>    5:   70  873 0.0015114832
#>   ---
#> 8291:    1 1560 0.0376633215
#> 8292:    1 1561 0.0376842272
#> 8293:    1 1562 0.0377005947
#> 8294:    1 1563 0.0377261376
#> 8295:    1 1564 0.0375441545
```

The `data.table` object `posterior` has three columns, cell id `cell, tile number` rid, `and posterior probability`pga'.

The map is created as follows:

```
# Edit settings
settings$titles["pga"] <- "Posterior probabilities"

# Create the plot
map_pga(rst,
        dt = posterior,
        cp = cp,
        region = region,
        cells = 3,
        interactive = FALSE,
        settings = settings)
```

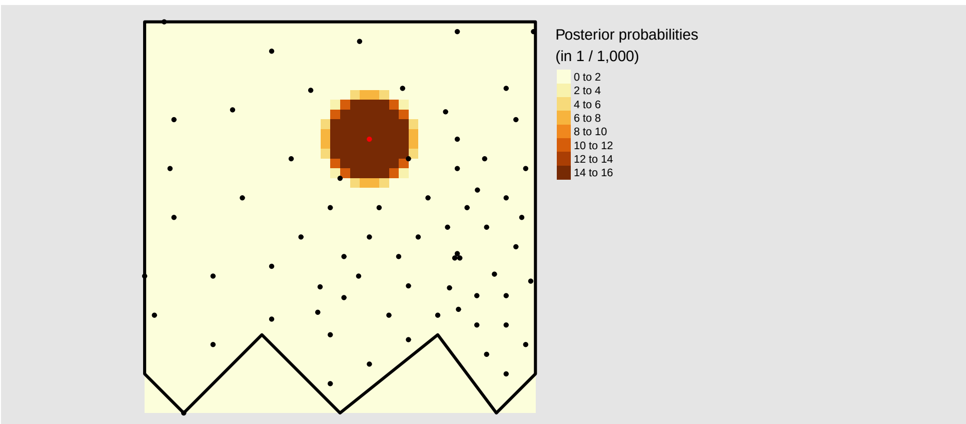

Figure 3.3 Map representation of the posterior location probabilities. Every black dot represents an omnidirectional BTS (multiple points indicate directional radio cells). The red dot denotes the BTS under visualization (number 3 in the code above).

4

Further developments

In our view, visualization, although a common element of software development in data analytics, stands as an essential tool in official statistical production both for dissemination and also, and increasingly more important, for analysis and development in early stages of the production chain (design included). It is for this reason that we decided in the work plan of this work package to devote a standalone deliverable for this element of production. The relevance of visualization for new data sources will steadily increase in the future.

Also, notice how this course of action is completely aligned with the fully modular and evolvable approach of the ESS Reference Methodological Framework substantiated in all preceding deliverables. In this document we have illustrated how key conceptual issues as the prior location probabilities, posterior location probabilities, and event location probabilities can be visualized with simple call executions. If new statistical methods are proposed and implemented in the future, these visualizations can still be used paying off for the modular and evolvable approach.

Furthermore, the visualization tools included in our R package `mobvis` also assist the statistician to design, decide, and understand the set of parameters underlying the computation of those probabilities.

In this line of thought, directions of improvement and completion for these visualization tools are:

- Integration of dynamical approaches to geolocation (such as e.g. the HMM approach in deliverable I.3).
- Integration of more radio wave propagation models for the computation of event location probabilities.

Notice, however, that any improvement regarding the visualisation should be preceded by a progress in the underlying statistical methodology and the necessary data access, thus requesting full coordination among different research teams. In this sense, the different initiatives within the European Statistical System beyond the present ESSnet project are providing this scenario so that in a hopefully near future mobile network data can be incorporated into the routinely production of official statistics under a common production framework.

Bibliography

Hijmans, R. J. (2020). *raster: Geographic Data Analysis and Modeling*. R package version 3.3-13.

Oancea, B., S. Barragán, D. Salgado, and L. Sanguiao (2020). WPI DeliverableI.4. Information Technologies - Some IT tools for the production of official statistics with mobile network data.

Oancea, B., M. Necula, D. Salgado, and L. Sanguiao (2019). WPIDeliverableI.2. Data Simulator - A simulator for network event data.

Pebesma, E. (2018). Simple features for r: Standardized support for spatial vector data. *The R Journal 10*(1), 439.

Radini, R., T. Tuoto, F. de Fausti, L. Valentino, R. M. Aracri, S. Hadam, S. Barragán, and D. Salgado (2020). WPI DeliverableI.5. Standards & Metadata - First proposed standards and metadata for the production of official statistics with mobile network data.

Salgado, D., L. Sanguiao, S. Barragán, B. Oancea, and M. Suarez-Castillo (2020). WPI DeliverableI.3. Methodology - A proposed production framework with mobile network data.

Tennekes, M. (2018). tmap: Thematic maps in R. *Journal of Statistical Software 84*(6), 1–39.

Tennekes, M. (2020a). *mobloc: Mobile phone location algorithms and tools*. R package version 0.5.

Tennekes, M. (2020b). *mobvis: Visualization of mobile phone location algorithm results*. R package version 0.1.0.

Tennekes, M., Y. A. Gootzen, and S. H. Shah (2020, May). A Bayesian approach to location estimation of mobile devices from mobile network operator data. resreport, Statistics Netherlands (CBS).