High-resolution ecogeographical variables for species distribution modelling describing Latvia, 2024

Andris Avotiņš

2025-10-14

Contents

Preface

Welcome! This book documents the geodata and processing workflows used to create ecogeographical variables (EGVs) for species distribution modelling in Latvia (2024).

This material has been developed to present the results of three projects implemented at the University of Latvia, which are deeply rooted in species distribution modeling, and, more importantly, to demonstrate and explain the work process and decisions made in order to ensure their repeatability and reproducibility. These projects are:

- The project "Preparation of a geospatial data layer covering existing protected areas for the implementation of the EU Biodiversity Strategy 2030" (No. 1-08/73/2023), funded by the Latvian Environmental Protection Fund Administration;

- Scientific research service project commissioned by AS "Latvijas valsts meži" (Latvian State Forests) "Improvement of the monitoring of the northern goshawk *Accipiter gentilis* and creation of a spatial model of habitat suitability" (Latvian State Forests document No. 5-5.5.1_000r_101_23_27_6);

- State research program "Development of research specified in the Biodiversity Priority Action Program" project "High-resolution quantification of biodiversity for nature conservation and management: HiQBioDiv" (VPP-VARAM-DABA-2024/1-0002).

The material was developed in R using {bookdown}. The data processing and analysis described in the content was mainly performed in R, and one of the main reasons for creating this material was to transfer the information necessary for reproducing the work using verified command lines. A desirable side effect is to promote openness and reproducibility in scientific practice and practical science.

- Repo: aavotins/HiQBioDiv_EGVs
- Cite as needed using `book/book.bib`.

About this material

This material **is not**:

- *an introduction to R or other programming languages*. On the contrary, it will be most useful to those who already understand how to use command lines. However, it will also be informative for other users regarding the approaches used;
- *a tutorial on geoprocessing*. This material summarizes the approaches that, at the time of its development, were known to the authors as the most effective (in terms of processing time, RAM and hard disk space, performance guarantees, and reliability), but they are certainly not the only ones possible;
- *copy/paste ready product*. Although the use and publication of command lines tends to be intended for these purposes, in a situation where large amounts of data and, at least in part, restricted access data are used for the work, this is simply not possible. However, by ensuring data availability and placement in accordance with the file structure of this project (availabe at `root/Data` or by forking template repository), the command lines will be repeatable without changes and will produce the same results.

This material **has been** prepared to provide a reproducible workflow, describing the decisions made and solutions implemented in the preparation of ecogeographical variables for species distribution (habitat suitability) modeling for biodiversity conservation planning.

For the most part, this material consists of:

- *explanatory text*, which is recognizable as text;
- *command lines*, which are hidden by default to make the text easier to read. The locations of the command lines can be identified by the "|> Code" visible on the left side of the page, just below this paragraph. Clicking on it will open the code area, where the text on a gray background is command lines, for example:

```
object=function(arguments1,arguments2,
path="./path/file/tree/object.extension")
# comment
```

In the example above, the first line creates an object ("object") that is the result of a function ("function()"). The function has three arguments ("arguments1", "arguments2" and "path") separated by commas (as with all function arguments in R). The third argument is the path in the file tree (it is on a new line but is a continuation of the function on the previous line, because the parentheses are not closed), which is indicated by an equal sign (and quotation marks) followed by this path (note the beginning "./", which indicates a relative path - the location in the file tree is relative to the project location).

The second line of the example above is a comment - everything after "#" is a comment. Anything in a command line before "#" must be an executable function or object. A comment can contain anything and be on the same line as an executable function (at the end of it).

Command lines are the most important part of this material for reproducibility. However, the person using them must ensure the availability of input data and maintain correct paths in the file tree.

Command lines can also be found in text, for example, `# comment as a command line ↪ in text`.

Sometimes I will refer to R packages in the text, I will put them in curly brackets, for example, {package}.

- *graphics* - occasional diagrams that describe the workflow or data characteristics and maps;
- *links to other resources*, especially to higher-level products and results created within the project, but also to input data, if it is publicly available. The results are intended for practical use.

Within reason, the material describes all data sets used and provides metadata related to ensuring reproducibility. Since not all data sets are freely available, they are not published as such, but in all cases information is provided on how they were obtained for the development of this project.

Outline

1. Terminology and acronyms
2. Utilities
3. Template files
4. Raw geodata
5. Geodata products
6. Ecogeographical variables
7. Data access

Chapter 1

Terminology and acronyms

Athough all georeferenced data can be considered *geodata*, in this material we use the following terms in the order listed below in our workflows:

- **raw geodata** - considered as raw data obtained for a harmonised description of the environment. This may include tables with coordinates, raster or vector data. It can be anything that has been or can be used to create *ecogeographical variables*, with or without slight processing.

- **geodata product** - processed *raw geodata* that have undegone heavy modifications, e.g. spatial overlays and combinations of different sets of *raw geodata*, and are used as *input data*. In this document, *geodata products* are categorical raster layers that match the *CRS* and the pixel locations of *input data*. When split by categories, they become *input data*. The processing step of creating *geodata products* is necessary when decisions about the order of spatial overlays are important. For example, in a high-resolution pixel, there can only be water or forest, if the edge between water and forest need to be calculated.

- **input data** or **input layers** - very-high resolution (multiple times higher than that used for *ecogeographical variables*) raster data that are the direct input for the creation of most of the *ecogeographical variables*. The creation of such layers is particularly useful alongside *geodata products*, as dealing with border misalignment or decisions regarding the order of spatial overlays, as well as simple geoprocessing, is much faster with raster data.

- **ecogeographical variables** (EGVs) - this is the final product of the workflow describing environment for statistical analysis (e.g. *species distribution modelling*). They are suitable also for publishing due to standadisation of the values. In other words, these are standardised landscape ecological variables in the for of high-resolution raster layers (we use 1 ha cells). Each layer contains values representing the environment within the cell footprint or a summary of focal neighbours. In our case, each layer is of quantitative data describing a natural quantity (e.g. timber volume, mean annual temperature), or quantified information of categories (e.g. the fraction of class's area in an analysis cell or some neighbourhood, the number of pixels creating an edge of a certain class or between two classes in the analysis cell or some neighbourhood). The values of each layer are standardised - from every cells value layers mean is subtracted and then every cells value is divided by layers root mean square error. Therefore, the values are more suitable for modelling, and the layers can be made publicly available as they do not directly provide exact sensitive information.

In this material, we use the term *species distribution modelling* as a more used term, that is synonymous with *ecological niche analysis* and *ecological niche modelling*.

Acronyms:

CRS - coordinate reference system

EGV - ecogeoraphical variables

SDM - species distribution modelling

SDMs - species distribution models

LAD - Rural support service

NDMI - normalized difference moisture index

NDVI - normalized difference vegetation index

NDWI - normalized difference water index

MVR - State Forest Service's stand level inventory database "Forest State Registry"

VMD - State Forest Service

Chapter 2

Utilities

This chapter provides a brief description of the utility functions used in this material. Most of these functions are packaged in the R package {egvtools}, which was created specifically for this work.

2.1 R package egvtools

{egvtools} provides a coherent set of wrappers and utilities that facilitate the reproducible and efficient creation of large-scale EGVs on real datasets. The package relies on robust building blocks — {terra}, {sf}, {sfarrow}, {exactextractr} and {whitebox} — and standardises input/output, naming conventions and multi-scale zonal statistics, ensuring that the pipelines are repeatable across machines and projects.

The package was developed for the project 'HiQBioDiv: High-resolution quantification of biodiversity for conservation and management', which was funded by the Latvian Council of Science (Ref. No. VPP-VARAM-DABA-2024/1-0002), to simplify our work and to facilitate the reproduction of our results. Five of the functions are strictly for replication, while others are useful for a wider audience.

Package can be installed from GitHub with:

```r
# install.packages("pak")
pak::pak("aavotins/egvtools")
```

or obtained as a Docker container with all the necessary system and software dependencies.

2.1.1 Reproduction only functions

These functions are small wrappers, that helps to recreate our working environments - template files and their locations in the file tree.

These functions are:

- `download_raster_templates()` — fetch template rasters from Zenodo repository and place them in user specified location on disk, or by default - the one we used. By default this functions links to version 2.0.0 of the dataset;
- `download_vector_templates()` - fetch template vector grids/points from Zenodo repository and place them in user specified location on disk, or by default - the one we used. By default this functions links to version 1.0.1 of the dataset;
- radius_function() — extracts summary statistics from raster layers using buffered polygon zones of multiple radii and rasterizes them onto a common template grid. Insternally connected to exact parts of the file names used in this project. If they are kept, can be used in other places.

2.1.2 General purpose functions

Each of those functions are small workflows themselves, that can be combined into larger workflows and used more widely, than for Latvia.

- `tile_vector_grid()` — tile template (vector) grid for chunked processing. The function internally is linked to our file naming convention. As long as it is maintained, function can be used to create tiled grid from any {sfarrow} parquet grid file;
- `tiled_buffers()` — precompute buffered tiles for multiple radii around points. The function internally is linked to our file naming convention. As long as it is maintained, function can be used to create tiled polygons with buffers around points from any {sfarrow} parquet grid file. There are three buffering modes: **dense** (buffers the best-matching pts100*.parquet (prefers pts100_sauzeme.parquet) for each tile by radii_dense (default: 500, 1250, 3000, 10000 m ensuring that every analysis grid cell has desired buffer. Computationally heavy in the following workflows), **sparse** (uses a file to radius mapping and is highly generalizable), and **specified** (the same as sparse, but with one single point file). **In our workflows we used the sparse mode with default mapping**;
- create_backgrounds() — a wrapper around `terra::ifel()` to build consistent background rasters. This function better guards coordinate reference system and how it is stored, while also guarding spatial cover, resolution, coordinate reference system, exact pixel matching, etc. Creation of layers with default background values is faster than recreating them several times in workflows preparing EGVs;

- polygon2input() — rasterize polygons to input layers. Handles only polygon data, other geometry types need to buffered. Rasterizes polygon/multipolygon sf data to a raster aligned to a template GeoTIFF. Rasterization targets a raster::RasterLayer built from the template (so grids normally match). Projection is optional (project_mode). Missing values are counted only over valid template cells. User may optionally restrict the result with a raster mask (restrict_to) using numeric values or bracketed range strings (e.g., "(0,5]", "[10,)"). Remaining NA cells can be filled by covering with a background raster (background_raster) or a constant (background_value). For large rasters, heavy steps (projection/mask/-cover) can stream to disk via terra_todisk=TRUE.

- input2egv() — normalize/align a fine-resolution input raster to a (coarser) EGV template, optionally cover missing values and/or fill gaps (IDW via Whitebox), and write the result to disk. Designed for large runs: fast gap counting (inside template footprint only), optional filling, tuned GDAL write options, and controlled terra memory/temp behavior.

- downscale2egv() — downscale coarse rasters to a template grid (CRS, resolution, extent), masks to the template footprint, and optionally: (1) fills NoData gaps using WhiteboxTools' IDW-based fill_missing_data, and (2) applies IDW smoothing to reduce blockiness from low-resolution inputs.

- distance2egv() — computes Euclidean distance (in map units) from cells matching a set of class values in an input raster to all cells of an EGV template grid, then writes a Float32 GeoTIFF aligned to the template. Designed to work with rasters produced by `polygon2input()`.

- landscape_function() — computes a {landscapemetrics} metric (default "lsm_l_shdi"), optionally with extra lm_args, that yields one value per zone and per input layer. Runs tile-by-tile (by tile_field), writes per-tile rasters, merges to final per-layer GeoTIFF(s), then performs gap analysis (NA count within the template footprint and optional maximum gap width) and optional IDW gap filling via WhiteboxTools. Returns a compact data.frame with per-layer stats and timing.

2.2 Other utility functions

Other handy functions repeatedly used, not included in {egvtools} are stored in `egvs02` ↪ `.02_UtilityFunctions.R` file, located in `Data/RScipts_final`.

- `ensure_multipolygons()` - rather agressive function to create `MULTIPOLYGON` geometries from `GEOMETRYCOLLECTION`

```
if(!require(sf)) {install.packages("sf"); require(sf)}
```

```
if(!require(gdalUtilities)) {install.packages("gdalUtilities"); require(
    ↪ gdalUtilities)}

ensure_multipolygons <- function(X) {
  library(sf)
  library(gdalUtilities)

  tmp1 <- tempfile(fileext = ".gpkg")
  tmp2 <- tempfile(fileext = ".gpkg")
  st_write(X, tmp1)
  ogr2ogr(tmp1, tmp2, f = "GPKG", nlt = "MULTIPOLYGON")
  Y <- st_read(tmp2)
  st_sf(st_drop_geometry(X), geom = st_geometry(Y))
}
```

Chapter 3

Templates and utilities

This chapter defines template files. They define the analysis space and ensure harmonisation of georeferenced data creation, and facilitate connection with other Latvian geodata.

3.1 Vector data

Baseline template (or reference) vector grid and point files are publically available at HiQBioDiv's Zenodo repository. Command lines and data used to create these files are documented at the HiQBioDiv main code repository's file.

The easiest way to obtain these files is to run determined function `download_vector_templates()` from {egvtools}.

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

download_vector_templates(
  url = "https://zenodo.org/api/records/14277114/files-archive",
  grid_dir = "./Templates/TemplateGrids",
  points_dir = "./Templates/TemplateGridPoints",
  gpkg_dir = "./Templates",
  overwrite = FALSE,
  quiet = FALSE
)
```

Once template vector data are downloaded and unarchived, they need to be tiled:

1. Analysis grid is tiled in `tks50km` pages

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

tile_vector_grid(
  grid_path = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  out_dir = "./Templates/TemplateGrids/tiles",
  tile_field = "tks50km",
  chunk_size = 50000L,
  overwrite = FALSE,
  quiet = FALSE
)
```

Expect to see warning: `This is an initial implementation of Parquet/Feather ↪ file support and geo metadata. This is tracking version 0.1.0 of the ↪ metadata (https://github.com/geopandas/geo-arrow-spec). This metadata ↪ specification may change and does not yet make stability promises. We do ↪ not yet recommend using this in a production setting unless you are able ↪ to rewrite your Parquet/Feather files.`

2. Point files are tiled and buffered. In workflows creating EGVs described in this document, we used "sparse" grid:

    - 500m buffers around every 100m grids center;
    - 1250m buffers around every 100m grids center;
    - 3000m buffers around every 300m grids center (to speed up neighbourhood analysis ~9 times, while loosing <0.001% of precission);
    - 10000m buffers around every 1000m grids center (to speed up neighbourhood analysis ~100 times, while loosing <0.001% of precission)

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

tiled_buffers(
  in_dir = "./Templates/TemplateGridPoints",
  out_dir = "./Templates/TemplateGridPoints/tiles",
  buffer_mode = "sparse",
  mapping_sparse = list("pts100_sauzeme.parquet" = c(500, 1250),
                        "pts300_sauzeme.parquet" =  3000,
                        "pts1000_sauzeme.parquet" = 10000),
  split_field = "tks50km",
  n_workers = 4,
  os_type = NULL,
  future_max_mem_gb = 4,
  overwrite = FALSE,
  quiet = FALSE
)
```

Expect to see warning: `This is an initial implementation of Parquet/Feather file support and geo metadata. This is tracking version 0.1.0 of the metadata (https://github.com/geopandas/geo-arrow-spec). This metadata specification may change and does not yet make stability promises. We do not yet recommend using this in a production setting unless you are able to rewrite your Parquet/Feather files.`

Apperance of file `pts300_r3000_NA.parquet`, i.e. without a tile number, is expected, due to slight mismatch of 300 m grid with the 50 km one.

3.2 Raster data

Baseline template (or reference) raster grid and point files are publically available at HiQBioDiv's Zenodo repository. Command lines and data used to create these files are documented at the HiQBioDiv main code repository's file.

The easiest way to obtain these files is to run determined function `download_raster_templates()` from {egvtools}.

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    (egvtools)}

download_raster_templates(
  url = "https://zenodo.org/api/records/14497070/files-archive",
  out_dir = "./Templates/TemplateRasters",
  overwrite = FALSE,
  quiet = FALSE
)
```

During EGV creation background covering to deal with missing values may be necessary. All the EGVs described in this document where such an excercise might be necessary can be considered quantities of ratio scale, therefore backgrounds with value `0` are created.

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    (egvtools)}

create_backgrounds(in_dir="./Templates/TemplateRasters/",
                   out_dir = "./Templates/TemplateRasters/",
                   background_value = 0,
                   out_prefix = "nulls_",
                   overwrite=TRUE)
```

Chapter 4

Raw geodata

This chapter describes raw geodata used and the preliminary processing conducted on them.

4.1 State Forest Service's State Forest Registry

The State Forest Service's Forest State Register database (ESRI file geodatabase), which compiles indicators and spatial data characterizing forest compartments (stand level inventory database), was received by the University of Latvia on January 7, 2024, to support study and research processes. The structure of the received database version corresponds to the Forest State Register Forest Inventory File Structure, but lowercase letters are used in field names.

After downloading, the CRS is guarded, geometries are checked and saved in geoparquet format.

Files are stored at `Geodata/2024/MVR/`.

```
# libs
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(gdalUtilities)) {install.packages("gdalUtilities"); require(
    ↪ gdalUtilities)}

# database
nog=read_sf("./Geodata/2024/MVR/VMD.gdb/",layer="Nogabali_pilna_datubaze")

# ensuring geometries
source("./RScripts_final/egvs02.02_UtilityFunctions.R")
nogabali <- ensure_multipolygons(nog)
```

```
# securing geometries
nogabali2 = nogabali[!st_is_empty(nogabali),,drop=FALSE] # 108 empty
    ↪ geometries
validity=st_is_valid(nogabali2)
table(validity) # 1733 invalid geometries
nogabali3=st_make_valid(nogabali2)

# transforming CRS
nogabali4=st_transform(nogabali3, crs=3059)

# saving
sfarrow::st_write_parquet(nogabali4, "./Geodata/2024/MVR/nogabali_2024janv.
    ↪ parquet")
```

4.2 Rural Support Service's information on declared fields

The Rural Support Service maintains regularly updated information on the open data portal. An archive (since 2015) is also available there, and the data sets that can be used contain the keyword "deklarētās platības".

After downloading files to `Geodata/2024/LAD/downloads/`, they are unzipped and read into R. it is checked, empty files are deleted and the rest are validated, and all individual files are combined into one, which is saved in geopackage and geoparquet formats at `Geodata/2024/LAD/`. At the end, downloaded files are unlinked.

```
# libs
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(gdalUtilities)) {install.packages("gdalUtilities"); require(
    ↪ gdalUtilities)}

# reading all files
faili=data.frame(celi=list.files("./Geodata/2024/LAD/downloads",full.names =
    ↪ TRUE))
dati=st_read(faili$celi[1])
for(i in 2:length(faili$celi)){
  nakosais=st_read(faili$celi[i])
  dati=bind_rows(dati,nakosais)
  print(nrow(dati))
}

# ensuring geometries
source("./RScripts_final/egvs02.02_UtilityFunctions.R")
```

```
nogabali <- ensure_multipolygons(nog)
dati2 <- ensure_multipolygons(dati)
dati3 = dati2[!st_is_empty(dati2),,drop=FALSE] # viss āīākrtb
table(st_is_valid(dati3))
dati4=st_make_valid(dati3)
table(st_is_valid(dati4))
dati5 <- ensure_multipolygons(dati4)
table(st_is_valid(dati5))

# saving output
st_write(dati5,"./Geodata/2024/LAD/Lauki_2024.gpkg",append = FALSE)
sfarrow::st_write_parquet(dati5,"./Geodata/2024/LAD/Lauki_2024.parquet")

# unlinking downloads
for(i in seq_along(faili$celi)){
  unlink(faili$celi[i])
}
```

4.3 Melioration Cadaster

The Land Improvement Cadastre Information System database was downloaded layer by layer from Geoserver. Geometries were tested and validated for each layer, and layers were all combined into a single geopackage file stored at `Geodata/2024/MKIS/`.

Initially, no additional processing was performed on this data. It was used to prepare Geodata products - both Terrain products and Landscape classification.

```
# libs
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(httr)) {install.packages("httr"); require(httr)}
if(!require(ows4R)) {install.packages("ows4R"); require(ows4R)}

# basis information ----
link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)
url$query <- list(service = "wfs",
                  #version = "2.0.0", # facultative
                  request = "GetCapabilities")
request <- build_url(url)
request
bwk_client <- WFSClient$new(link,
                            serviceVersion = "2.0.0")
bwk_client
bwk_client$getFeatureTypes(pretty = TRUE)
```

```
# aizsargdambji ----

bwk_client$getFeatureTypes(pretty = TRUE)
url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_dam")
request <- build_url(url)
aizsargdambji <- read_sf(request)
aizsargdambji = aizsargdambji %>% st_set_crs(st_crs(3059))
aizsargdambji=st_cast(aizsargdambji,"MULTILINESTRING")

ggplot(aizsargdambji)+geom_sf()

table(st_is_valid(aizsargdambji))

write_sf(aizsargdambji,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="Aizsargdambji",
         append=FALSE)
rm(aizsargdambji)

# ādabisks ūdensteces ----

bwk_client$getFeatureTypes(pretty = TRUE)
url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_watercourses")
request <- build_url(url)

DabiskasUdensteces <- read_sf(request)
DabiskasUdensteces = DabiskasUdensteces %>% st_set_crs(st_crs(3059))
DabiskasUdensteces=st_cast(DabiskasUdensteces,"MULTILINESTRING")

ggplot(DabiskasUdensteces)+geom_sf()

table(st_is_valid(DabiskasUdensteces))

write_sf(DabiskasUdensteces,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="DabiskasUdensteces",
         append=FALSE)
rm(DabiskasUdensteces)



# dambju piketi ----
```

```
bwk_client$getFeatureTypes(pretty = TRUE)
url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_dampicket")
request <- build_url(url)

DambjuPiketi <- read_sf(request)
DambjuPiketi = DambjuPiketi %>% st_set_crs(st_crs(3059))
DambjuPiketi=st_cast(DambjuPiketi,"POINT")

ggplot(DambjuPiketi)+geom_sf()

table(st_is_valid(DambjuPiketi))

write_sf(DambjuPiketi,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="DambjuPiketi",
         append=FALSE)
rm(DambjuPiketi)


# drenas ----

bwk_client$getFeatureTypes(pretty = TRUE)

base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_drainpipes"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_Drenas"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)
```

```
  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("MULTILINESTRING")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

Drenas_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                   layer="temp_Drenas")
Drenas_all2 = Drenas_all[!st_is_empty(Drenas_all),,drop=FALSE] # 1
table(st_is_valid(Drenas_all2))


write_sf(Drenas_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="Drenas",
         append=FALSE)
rm(list=ls())




# drenu kolektori ----



bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam
bwk_client$getFeatureTypes(pretty = TRUE)
url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_draincollectors",
```

```
                  count=1)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam

# skaitam
url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_draincollectors",
                  resultType="hits")
request <- build_url(url)
result <- GET(request)
parsed <- xml2::as_list(content(result, "parsed"))
n_features <- attr(parsed$FeatureCollection, "numberMatched")
n_features


# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_draincollectors"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_DrenuKolektori"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING
    chunk <- chunk %>%
```

```
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("MULTILINESTRING")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

DrenuKolektori_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                           layer="temp_DrenuKolektori")
DrenuKolektori_all2 = DrenuKolektori_all[!st_is_empty(DrenuKolektori_all),,
    ↪ drop=FALSE] # 1
table(st_is_valid(DrenuKolektori_all2))


write_sf(DrenuKolektori_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="DrenuKolektori",
         append=FALSE)
rm(list=ls())




# āždrenas ītkla ūbves ----

link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
```

```
                  typename = "zmni:zmni_networkstructures",
                  count=1)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_networkstructures"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_DrenazasTiklaBuves"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("POINT")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
```

```
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

DrenazasTiklaBuves_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                               layer="temp_DrenazasTiklaBuves")
DrenazasTiklaBuves_all2 = DrenazasTiklaBuves_all[!st_is_empty(
    ↪ DrenazasTiklaBuves_all),,drop=FALSE] # 0
table(st_is_valid(DrenazasTiklaBuves_all2))


write_sf(DrenazasTiklaBuves_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="DrenazasTiklaBuves",
         append=FALSE)
rm(list=ls())




# āgrvji -----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_ditches",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam
```

```
# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_ditches"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_Gravji"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("MULTILINESTRING")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

Gravji_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
```

```
                   layer="temp_Gravji")
Gravji_all2 = Gravji_all[!st_is_empty(Gravji_all),,drop=FALSE] # 0
table(st_is_valid(Gravji_all2))


write_sf(Gravji_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="Gravji",
         append=FALSE)
rm(list=ls())





# hidrometriskie ņpostei ----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_hydropost",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_hydropost"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./IevadesDati/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_HidrometriskiePosteni"
i <- 0

repeat {
```

```
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("POINT")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

HidrometriskiePosteni_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                                  layer="temp_HidrometriskiePosteni")
HidrometriskiePosteni_all2 = HidrometriskiePosteni_all[!st_is_empty(
    ↪ HidrometriskiePosteni_all),,drop=FALSE] # 0
table(st_is_valid(HidrometriskiePosteni_all2))


write_sf(HidrometriskiePosteni_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="HidrometriskiePosteni",
         append=FALSE)
```

```
rm(list=ls())


# liela diametra kolektori ----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_bigdraincollectors",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_bigdraincollectors"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_LielaDiametraKolektori"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
```

```
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("MULTILINESTRING")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

LielaDiametraKolektori_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                                   layer="temp_LielaDiametraKolektori")
LielaDiametraKolektori_all2 = LielaDiametraKolektori_all[!st_is_empty(
    ↪ LielaDiametraKolektori_all),,drop=FALSE] # 0
table(st_is_valid(LielaDiametraKolektori_all2))


write_sf(LielaDiametraKolektori_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="LielaDiametraKolektori",
         append=FALSE)
rm(list=ls())




# piketi ----



link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)
```

```r
url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_stateriverspickets",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_stateriverspickets"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_Piketi"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break
```

```
    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("POINT")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

Piketi_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                   layer="temp_Piketi")
Piketi_all2 = Piketi_all[!st_is_empty(Piketi_all),,drop=FALSE] # 0
table(st_is_valid(Piketi_all2))


write_sf(Piketi_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="Piketi",
         append=FALSE)
rm(list=ls())


# polderu ūņsku stacijas -----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
```

```
                  typename = "zmni:zmni_polderpumpingstation",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_polderpumpingstation"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_PolderuSuknuStacijas"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("POINT")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
```

```
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

PolderuSuknuStacijas_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                                 layer="temp_PolderuSuknuStacijas")
PolderuSuknuStacijas_all2 = PolderuSuknuStacijas_all[!st_is_empty(
    ↪ PolderuSuknuStacijas_all),,drop=FALSE] # 0
table(st_is_valid(PolderuSuknuStacijas_all2))


write_sf(PolderuSuknuStacijas_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="PolderuSuknuStacijas",
         append=FALSE)
rm(list=ls())




# polderu teritorijas -----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_polderterritory",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam

geometrijas=st_set_crs(geometrijam,st_crs(3059))

library(gdalUtilities)
```

```
ensure_multipolygons <- function(X) {
  tmp1 <- tempfile(fileext = ".gpkg")
  tmp2 <- tempfile(fileext = ".gpkg")
  st_write(X, tmp1)
  ogr2ogr(tmp1, tmp2, f = "GPKG", nlt = "MULTIPOLYGON")
  Y <- st_read(tmp2)
  st_sf(st_drop_geometry(X), geom = st_geometry(Y))
}
poligoni <- ensure_multipolygons(geometrijas)
PolderuTeritorijas_all2 = poligoni[!st_is_empty(poligoni),,drop=FALSE] # 0
table(st_is_valid(PolderuTeritorijas_all2))


# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_polderterritory"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_PolderuTeritorijas"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("MULTIPOLYGON")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
```

```
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)


PolderuTeritorijas_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                               layer="temp_PolderuTeritorijas")
PolderuTeritorijas_all2 = PolderuTeritorijas_all[!st_is_empty(
    ↪ PolderuTeritorijas_all),,drop=FALSE] # 0
table(st_is_valid(PolderuTeritorijas_all2))


write_sf(PolderuTeritorijas_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="PolderuTeritorijas",
         append=FALSE)
rm(list=ls())


# sateces baseini ----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_catchment",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
```

```
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_catchment"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_SatecesBaseini"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code))

    ensure_multipolygons <- function(X) {
      tmp1 <- tempfile(fileext = ".gpkg")
      tmp2 <- tempfile(fileext = ".gpkg")
      st_write(X, tmp1)
      ogr2ogr(tmp1, tmp2, f = "GPKG", nlt = "MULTIPOLYGON")
      Y <- st_read(tmp2)
      st_sf(st_drop_geometry(X), geom = st_geometry(Y))
    }
    chunk <- ensure_multipolygons(chunk)


    # Write chunk to GeoPackage (append mode after first)
    st_write(
```

```
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)


SatecesBaseini_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                           layer="temp_SatecesBaseini")
SatecesBaseini_all2 = SatecesBaseini_all[!st_is_empty(SatecesBaseini_all),,
    ↪ drop=FALSE] # 0
table(st_is_valid(SatecesBaseini_all2))

SatecesBaseini_all3=st_make_valid(SatecesBaseini_all2)
table(st_is_valid(SatecesBaseini_all3))
SatecesBaseini_all3

write_sf(SatecesBaseini_all3,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="SatecesBaseini",
         append=FALSE)
rm(list=ls())


# savienojumi ----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_connectionpoints",
```

```
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_connectionpoints"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_Savienojumi"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code))

    chunk=st_cast(chunk,"POINT")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
```

```
    )

    i <- i + 1
  }, silent = TRUE)
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)


Savienojumi_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                        layer="temp_Savienojumi")
Savienojumi_all2 = Savienojumi_all[!st_is_empty(Savienojumi_all),,drop=FALSE]
    ↪ # 0
table(st_is_valid(Savienojumi_all2))


write_sf(Savienojumi_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="Savienojumi",
         append=FALSE)
rm(list=ls())


# valsts īnozme ūdensnotekas -----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_statecontrolledrivers",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam
```

```
# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_statecontrolledrivers"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_ValstsNozimesUdensnotekas"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code))

    chunk=st_cast(chunk,"MULTILINESTRING")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)
```

```
ValstsNozimesUdensnotekas_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.
    ↪ gpkg",
                                      layer="temp_ValstsNozimesUdensnotekas")
ValstsNozimesUdensnotekas_all2 = ValstsNozimesUdensnotekas_all[!st_is_empty(
    ↪ ValstsNozimesUdensnotekas_all),,drop=FALSE] # 0
table(st_is_valid(ValstsNozimesUdensnotekas_all2))


write_sf(ValstsNozimesUdensnotekas_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="ValstsNozimesUdensnotekas",
         append=FALSE)
rm(list=ls())


# zmni ģreions ----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_zmniregion",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam


library(gdalUtilities)


# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_zmniregion"
crs_code <- 3059
chunk_size <- 100000
```

```r
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_ZMNIRegions"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code))

    ensure_multipolygons <- function(X) {
      tmp1 <- tempfile(fileext = ".gpkg")
      tmp2 <- tempfile(fileext = ".gpkg")
      st_write(X, tmp1)
      ogr2ogr(tmp1, tmp2, f = "GPKG", nlt = "MULTIPOLYGON")
      Y <- st_read(tmp2)
      st_sf(st_drop_geometry(X), geom = st_geometry(Y))
    }
    chunk <- ensure_multipolygons(chunk)


    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
```

```
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)


ZMNIRegions_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                        layer="temp_ZMNIRegions")
ZMNIRegions_all2 = ZMNIRegions_all[!st_is_empty(ZMNIRegions_all),,drop=FALSE]
    ↪  # 0
table(st_is_valid(ZMNIRegions_all2))


write_sf(ZMNIRegions_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="ZMNIRegions",
         append=FALSE)
rm(list=ls())





# ūdensnotekas (ānovadrvji) -----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_waterdrainditches",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
```

```
type_name <- "zmni:zmni_waterdrainditches"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_UdensnotekasNovadgravji"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code))

    chunk=st_cast(chunk,"MULTILINESTRING")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)
```

```
UdensnotekasNovadgravji_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
    ↪ ,
                                    layer="temp_UdensnotekasNovadgravji")
UdensnotekasNovadgravji_all2 = UdensnotekasNovadgravji_all[!st_is_empty(
    ↪ UdensnotekasNovadgravji_all),,drop=FALSE] # 0
table(st_is_valid(UdensnotekasNovadgravji_all2))


write_sf(UdensnotekasNovadgravji_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="UdensnotekasNovadgravji",
         append=FALSE)
rm(list=ls())




# ūdensnoteku un āgrvju piketi ----



link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_ditchpicket",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_ditchpicket"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
```

```
layer_name <- "temp_UdensnotekuNovadgravjuPiketi"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code)) %>%
      st_cast("POINT")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)

UdensnotekuNovadgravjuPiketi_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.
    ↪ gpkg",
                                         layer="temp_
    ↪ UdensnotekuNovadgravjuPiketi")
UdensnotekuNovadgravjuPiketi_all2 = UdensnotekuNovadgravjuPiketi_all[!st_is_
    ↪ empty(UdensnotekuNovadgravjuPiketi_all),,drop=FALSE] # 0
table(st_is_valid(UdensnotekuNovadgravjuPiketi_all2))
```

```
write_sf(UdensnotekuNovadgravjuPiketi_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="UdensnotekuNovadgravjuPiketi",
         append=FALSE)
rm(list=ls())



# ūčdensteu asis ----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_stateriversline",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam



# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_stateriversline"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_UdenstecuAsis"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
```

```
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )

  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code))

    chunk=st_cast(chunk,"MULTILINESTRING")

    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)


UdenstecuAsis_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg",
                          layer="temp_UdenstecuAsis")
UdenstecuAsis_all2 = UdenstecuAsis_all[!st_is_empty(UdenstecuAsis_all),,drop=
     ↪ FALSE] # 0
table(st_is_valid(UdenstecuAsis_all2))


write_sf(UdenstecuAsis_all2,
         "./Geodata/2024/MKIS/MKIS_2025.gpkg",
         layer="UdenstecuAsis",
         append=FALSE)
rm(list=ls())
```

```
# ūšdensteu ūdens virsmas laukumi ----


link="https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
url=parse_url(link)

url$query <- list(service = "wfs",request = "GetCapabilities")
request <- build_url(url)
bwk_client <- WFSClient$new(link,serviceVersion = "2.0.0")

bwk_client$getFeatureTypes(pretty = TRUE)

# geometrijam

url$query <- list(service = "wfs",
                  request = "GetFeature",
                  srsName="EPSG:3059",
                  typename = "zmni:zmni_stateriverspolygon",
                  count=100)
request <- build_url(url)

geometrijam <- read_sf(request)
geometrijam


# ālejupieldei
base_url <- "https://lvmgeoserver.lvm.lv/geoserver/zmni/ows?"
type_name <- "zmni:zmni_stateriverspolygon"
crs_code <- 3059
chunk_size <- 100000
gpkg_path <- "./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
layer_name <- "temp_UdenstecuVirsmasLaukumi"
i <- 0

repeat {
  message("Fetching features ", i * chunk_size + 1, " to ", (i + 1) * chunk_
    ↪ size, "...")

  query <- list(
    service = "WFS",
    version = "2.0.0",
    request = "GetFeature",
    typename = type_name,
    srsName = paste0("EPSG:", crs_code),
    count = chunk_size,
    startIndex = i * chunk_size
  )
```

```
  req_url <- modify_url(base_url, query = query)

  try({
    chunk <- read_sf(req_url)
    if (nrow(chunk) == 0) break

    # Set CRS and cast to MULTILINESTRING, POINT, MULTIPOLYGON
    chunk <- chunk %>%
      st_set_crs(st_crs(crs_code))

    ensure_multipolygons <- function(X) {
      tmp1 <- tempfile(fileext = ".gpkg")
      tmp2 <- tempfile(fileext = ".gpkg")
      st_write(X, tmp1)
      ogr2ogr(tmp1, tmp2, f = "GPKG", nlt = "MULTIPOLYGON")
      Y <- st_read(tmp2)
      st_sf(st_drop_geometry(X), geom = st_geometry(Y))
    }
    chunk <- ensure_multipolygons(chunk)


    # Write chunk to GeoPackage (append mode after first)
    st_write(
      chunk,
      dsn = gpkg_path,
      layer = layer_name,
      append = i != 0,
      quiet = FALSE
    )

    i <- i + 1
  }, silent = TRUE)
  Sys.sleep(0.5)
}

message("All chunks written to ", gpkg_path, " in layer ", layer_name)


UdenstecuVirsmasLaukumi_all=st_read("./Geodata/2024/MKIS/temp_MKIS_2025.gpkg"
    ↪ ,
                                    layer="temp_UdenstecuVirsmasLaukumi")
UdenstecuVirsmasLaukumi_all2 = UdenstecuVirsmasLaukumi_all[!st_is_empty(
    ↪ UdenstecuVirsmasLaukumi_all),,drop=FALSE] # 0
table(st_is_valid(UdenstecuVirsmasLaukumi_all2))

UdenstecuVirsmasLaukumi_all3=st_make_valid(UdenstecuVirsmasLaukumi_all2)
table(st_is_valid(UdenstecuVirsmasLaukumi_all3))

write_sf(UdenstecuVirsmasLaukumi_all3,
```

```
        "./Geodata/2024/MKIS/MKIS_2025.gpkg",
        layer="UdenstecuVirsmasLaukumi",
        append=FALSE)
rm(list=ls())
```

4.4 TopographicMap

To ensure the research process at the University of Latvia, the third (completed by January 1, 2018) and fourth (unfinished) versions of the Latvian Geospatial Information Agency's topographic map M:10000 vector geodatabase were received. The most recent version is available for public viewing, but access to vector data is restricted.

For the purposes of this project, the ESRI geodatabase has been converted to a geopackage file. As part of the file format change, geometries (empty, their validity checked and corrected where necessary) and coordinate system have been checked.

Files were stored at `Geodata/2024/TopographicMap/`.

After dealing with each database seperately, layers used in this project were combined, preffering the most timely per mapping page. These layers are:

- `bride_L`, describing bridges as lines;
- `bridge_P`, describing bridges as points;
- `hidro_A`, describing waterbodies as polygons;
- `hidro_L`, describing ditches and small rivers as lines;
- `landus_A`, describing LULC as polygons;
- `road_A`, describing larger roads as polygons;
- `road_L`, describing different including very small and unused roads as lines;
- `swamp_A`, describing bogs as polygons;
- `flora_L`, describing linear tree and shrub formations.

```
# libs ----
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(openxlsx)) {install.packages("openxlsx"); require(openxlsx)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# v4 ----
```

```
slani_v4=st_layers("./Geodata/2024/TopographicMap/Latvija_LKS92_v4_20250703.
    ↪ gdb/")
write.xlsx(slani_v4,"./Geodata/2024/TopographicMap/slani_v4partial.xlsx")

slani_v4$geometrijai=as.character(slani_v4$geomtype)
table(slani_v4$geometrijai)

slani_v4$geometrijai2=ifelse(slani_v4$geometrijai=="3D Point","POINT",
                                   ifelse(slani_v4$geometrijai=="Multi
    ↪ Polygon","MULTIPOLYGON",
                                          ifelse(slani_v4$geometrijai=="3D
    ↪ Multi Line String","MULTILINESTRING",
                                                 ifelse(slani_v4$geometrijai
    ↪ =="3D Multi Polygon","MULTIPOLYGON",NA))))

slani4x=data.frame(name=slani_v4$name,
                   geometrija=slani_v4$geometrijai2)

ciklam4x=levels(factor(slani4x$name))
for(i in seq_along(ciklam4x)){
  print(i)
  sakums=Sys.time()
  nosaukums=ciklam4x[i]
  objekts=slani4x %>%
    filter(name==nosaukums)
  print(nosaukums)
  slanis=read_sf("./Geodata/2024/TopographicMap/topo10v4/Latvija_LKS92_v4_
    ↪ 20250703.gdb/",layer=nosaukums)
  slanisZM=st_zm(slanis)
  slanis2=st_cast(slanisZM,to=objekts$geometrija)
  write_sf(slanis2,"./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg"
    ↪ ,layer=nosaukums,append=FALSE)
  ilgums=Sys.time()-sakums
  print(ilgums)
}



# v3 ----

slani_v3=st_layers("./Geodata/2024/TopographicMap/Latvija_LKS92_v3_pilnais.
    ↪ gdb/")
write.xlsx(slani_v3,"./Geodata/2024/TopographicMap/slani_v3.xlsx")

slani_v3$geometrijai=as.character(slani_v3$geomtype)
table(slani_v3$geometrijai)

slani_v3$geometrijai2=ifelse(slani_v3$geometrijai=="3D Point","POINT",
                                   ifelse(slani_v3$geometrijai=="Multi
    ↪ Polygon","MULTIPOLYGON",
```

```
                                          ifelse(slani_v3$geometrijai=="3D
    ↪ Multi Line String","MULTILINESTRING",
                                                 ifelse(slani_v3$geometrijai
    ↪ =="3D Multi Polygon","MULTIPOLYGON",
                                                        ifelse(slani_v3$
    ↪ geometrijai=="Point","POINT",
                                                               ifelse(slani_
    ↪ v3$geometrijai=="Multi Line String","MULTILINESTRING",
                                                                      ifelse(
    ↪ slani_v3$geometrijai=="3D Measured Point","POINT",NA))))))

slani3x=data.frame(name=slani_v3$name,
                   geometrija=slani_v3$geometrijai2)

ciklam3x=levels(factor(slani3x$name))
for(i in seq_along(ciklam3x)){
  print(i)
  sakums=Sys.time()
  nosaukums=ciklam3x[i]
  objekts=slani3x %>%
    filter(name==nosaukums)
  print(nosaukums)
  slanis=read_sf("./Geodata/2024/TopographicMap/Latvija_LKS92_v3_pilnais.gdb/
    ↪ ",layer=nosaukums)
  slanisZM=st_zm(slanis)
  slanis2=st_cast(slanisZM,to=objekts$geometrija)
  write_sf(slanis2,"./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer=
    ↪ nosaukums,append=FALSE)
  ilgums=Sys.time()-sakums
  print(ilgums)
}

# combination ----
st_layers("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg")

pages4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="Topo10_lapas")
pages4_united=st_union(pages4)
ggplot(pages4_united)+geom_sf()

# landus_A
landus_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ landus_A")
landus_not4=st_difference(landus_3,pages4_united)
landus_not4=landus_not4 %>%
  dplyr::select(FNAME,FCODE)
landus_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="landus_A")
landus_4=landus_4 %>%
  dplyr::select(FNAME,FCODE)
```

```
landus_new=rbind(landus_not4,landus_4)
sfarrow::st_write_parquet(landus_new,"./Geodata/2024/TopographicMap/LandusA_
    ↪ COMB.parquet")

# bridge_L
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ bridge_L")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="bridge_L")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/BridgeL_
    ↪ COMB.parquet")


# bridge_P
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ bridge_P")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="bridge_P")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/BridgeP_
    ↪ COMB.parquet")


# hidro_A
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ hidro_A")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="hidro_A")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/HidroA_COMB
```

```
    ↪ .parquet")


# hidro_L
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ hidro_L")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="hidro_L")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/HidroL_COMB
    ↪ .parquet")


# road_A
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ road_A")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="road_A")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/RoadA_COMB.
    ↪ parquet")


# road_L
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ road_L")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="road_L")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/RoadL_COMB.
    ↪ parquet")
```

```
# swamp_A
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ swamp_A")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="swamp_A")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/SwampA_COMB
    ↪ .parquet")


# flora_L
data_3=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v3.gpkg",layer="
    ↪ flora_L")
data_not4=st_difference(data_3,pages4_united)
data_not4=data_not4 %>%
  dplyr::select(FNAME,FCODE)
data_4=st_read("./Geodata/2024/TopographicMap/LGIAtopo10K_v4partial.gpkg",
    ↪ layer="flora_L")
data_4=data_4 %>%
  dplyr::select(FNAME,FCODE)

data_new=rbind(data_not4,data_4)
sfarrow::st_write_parquet(data_new,"./Geodata/2024/TopographicMap/FloraL_COMB
    ↪ .parquet")
```

4.5 Corine Land Cover 2018

Corine Land Cover is publicly available geodata that characterizes land cover and land use (LULC) across Europe over a long period of time using a generally consistent (comparable) methodology (https://land.copernicus.eu/content/corine-land -cover-nomenclature-guidelines/docs/pdf/CLC2018_Nomenclature_illustrated_guide_20190510.pdf), providing results for individual years - 1990, 2000, 2006, 2012, 2018 (https://land.copernicus.eu/en/products/corine-land-cover). Although the dataset has a coarse resolution – the mapping unit is 25 ha areas that are at least 100 m wide – it provides sufficient information for general use, such as workflow testing and observation filtering. This project uses data from 2018.

The downloaded data set has been transformed into the Latvian coordinate system (EPSG:3059), and the file format has been changed to geoparquet to facilitate and speed up further work. As part of the file format change, geometries (empty, validity) have been checked.

Data are stored at `Geodata/2024/CLC/`.

```
# libs ----
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}

# downloaded data
clcLV=st_read("./Geodata/2024/CLC/clcLV.gpkg",layer="clcLV")

# šātuks ģeometrijas
clcLV2 = clcLV[!st_is_empty(clcLV),,drop=FALSE] # OK

# ģeometriju ēšvalidana
validity=st_is_valid(clcLV2)
table(validity) # 3 non-valid
clcLV3=st_make_valid(clcLV2)

# ākoordintu ēsistma
clcLV3=st_transform(clcLV3,crs=3059)

# āšsaglabana
sfarrow::st_write_parquet(clcLV3, "./Geodata/2024/CLC/CLC_LV_2018.parquet")
```

4.6 Publicly available LVM data

Latvian State Forests geospatial data on forest infrastructure and its description. The following data sets were used in the project: - roads: - forest roads; - forest roads to be developed; - turning areas; - changeover areas; - driveways; - drainage systems: - ditches; - drainage systems; - renovated drainage facilities. Initially, no additional processing of this data was performed. It was used to prepare geodata products (more specifically, Landscape classification).

Data were downloaded to `Geodata/2024/LVM_OpenData`

4.7 Soil data

Directory `Geodata/2024/Soils/` contains various soil related datasets that need to be combined (soil texture) or can be used individually (soil chemistry). These datasets and their location in the filetree are documented in following subchapters.

4.7.1 Soil chemistry

Data on soil chemistry are obtained from European Soil Data Centre's European Soil database (Panagos et al., 2022). Dataset decribing soil chemistry is derived from LUCAS 2009/2012 topsoil data. There are several chemical properties available with download, however not all of them are experts chosen for SDM, therefore not used further in this work:

- "P": used;
- "N": used;
- "K": used;
- "CEC": not used;
- "CN": used;
- "pH_CaCl": not used;
- "ph_H2o_ration_ph_CaCl": not used;
- "pH_H2O": used;
- "CaCO3": used.

Files were downloaded to `Geodata/2024/Soils/ESDAC/chemistry/` and no preprocessing was carried out.

4.7.2 Soil texture: Europe

Data on soil texture are obtained from European Soil Data Centre's European Soil database (Panagos et al., 2022). Dataset is available as European Soil Database v2 Raster Library 1kmx1km. There are several properties available with download, `TXT` was used to create soil texture product. Files were downloaded to `eodata/2024/Soils/`
`↪ ESDAC/texture/`.

During the preprocessing (code below) layer was projected to match 10 m template with "near" as interpolation method, value `0` substituted with `NA` and masked and cropped to template. Result was saved for further processing.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# Template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
```

```
# ESDAC texture ----

sdTEXT=rast("./Geodata/2024/Soils/ESDAC/texture/SoilDatabaseV2_raster/ESDB-
    ↪ Raster-Library-1k-GeoTIFF-20240507/TEXT/TEXT.tif")
plot(sdTEXT)

sdTEXT=project(sdTEXT,template10,method="near")
plot(sdTEXT)

sdTEXT=subst(sdTEXT,0,NA)
plot(sdTEXT)

sdTEXT2=mask(sdTEXT,template10,
             filename="./RasterGrids_10m/2024/SoilTXT_ESDAC.tif",
             overwrite=TRUE)
plot(sdTEXT2)
```

4.7.3 Soil texture: Farmland

Topsoil characteristics in Latvia were mapped in the mid-20th century, almost exclusively in farmlands. With time, data were digitised and combined with some other information creating artefacts. Therefore preprocessing was necessary. The version we used was obtained form project "GOODWATER" C1D1_Deliverable_R2.

File is stored at `Geodata/2024/Soils/TopSoil_LV/`.

Preprocessing included:

- reclassification:
  - we coded as `clay` (3) following labels from field `GrSast` - "M","M1","Mp","M2","sM1","sMp1";
  - we coded as `silt` (2) following labels from field `GrSast` - "sM", "sMp", "M2", "sM2", "sMp2", "sM3", "sMp3";
  - we coded as `sand` (1) following labels from field `GrSast` - "mS", "mSp", "S", "sS", "iS", "Gr", "mGr", "D";
  - we coded as `organic` (4) following labels from field `GrSast` - "l", "vd", "vj", "n","T";
  - left others as unclassified.
- coordinate transformation to epsg:3059;
- invsestigated resulting layer looking for anomalies by scrolling in interactive GIS. Investigations led to exclusion of land parcels from 200 ha.

- rasterization to 10 m template with highest class code prevailing.

```r
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# Template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# Farmland soil texture ----

augsnes=st_read("./Geodata/2024/Soils/TopSoil_LV/soil.gpkg",layer="soilunion"
    ↪ )

# calculate parcels area
augsnes$platiba_ha=as.numeric(st_area(augsnes))/10000

# only parcels with existing information on texture
tuksas=augsnes %>%
  filter(GrSast=="")

# classification
clay=c("M","M1","Mp","M2","sM1","sMp1")
silt=c("sM", "sMp", "M2", "sM2", "sMp2", "sM3", "sMp3")
sand=c("mS", "mSp", "S", "sS", "iS", "Gr", "mGr", "D")
peat=c("l", "vd", "vj", "n","T")
augsnes=augsnes %>%
  mutate(grupas=case_when(GrSast %in% sand~"Sand",
                          GrSast %in% silt~"Silt",
                          GrSast %in% clay~"Clay",
                          GrSast %in% peat~"organika",
                          .default=NA)) %>%
  mutate(grupas_num=case_when(GrSast %in% sand~"1",
                              GrSast %in% silt~"2",
                              GrSast %in% clay~"3",
                              GrSast %in% peat~"4",
                              .default=NA))

# crs
augsnes_3059=st_transform(augsnes,crs=3059)

# only existing texture classification
augsnes_3059=augsnes_3059 %>%
  filter(!is.na(grupas_num))

# parcels up to 200 ha
augsnes_3059small=augsnes_3059 %>%
  filter(!is.na(grupas_num)) %>%
```

```
  filter(platiba_ha<200)

# rasterization
virsaugsnem2=rasterize(augsnes_3059small,template10,field="grupas_num",fun="
    ↪ max",
                       filename="./RasterGrids_10m/2024/SoilTXT_topSoilLV.tif
    ↪ ",
                       overwrite=TRUE)
plot(virsaugsnem2)
```

4.7.4 Soil texture: Quaternary

Data on Quaternary Geology are digitised and stored by University of Latvia Geology group.

File is stored at `Geodata/2024/Soils/QuaternaryGeology_LV/`.

Preprocessing included:

- reclassification:
  - we coded as `sand` (1) following values from field `Litologija` - “smilts”, “smilts_aleiritiska”, “smilts_dunjaina”, “smilts_grants”, “smilts_grants_oli”, “smilts_grants_oli_aleirits”, “smilts_kudraina”, “smilts_videjgraudaina, malsmilts”, “smilts_videjgraudaina”~“Sand”;
  - we coded as `silt` (2) following values from field `Litologija` ↪ - “aleirits”, “aleirits_malains”, “morena”, “smilts_aleirits_mals”, “smilts_aleirits_sapropelis”, “smilts_malaina_dazadgraudaina, malsmilts”;
  - we coded as `clay` (3) following values from field `Litologija` - “mals”, “mals_aleiritisks”;
  - we coded as `organic` (4) following values from field `Litologija` - “dunjas”, “kudra”;
- coordinate transformation to epsg:3059;
- rasterization to 10 m template with highest class code prevailing.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# Template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
```

```r
# Quarternary geology ----

kvartars=sfarrow::st_read_parquet("./Geodata/2024/Soils/QuaternaryGeology_LV/
    ↪ Kvartargeologija.parquet")

# reclassification
kvartars=kvartars %>%
  mutate(grupas = case_when(Litologija=="aleirits"~"Silt",
                            Litologija=="aleirits_malains"~"Silt",
                            Litologija=="dunjas"~"organika",
                            Litologija=="kudra"~"organika",
                            Litologija=="mals"~"Clay",
                            Litologija=="mals_aleiritisks"~"Clay",
                            Litologija=="morena"~"Silt",
                            Litologija=="smilts"~"Sand",
                            Litologija=="smilts_aleiritiska"~"Sand",
                            Litologija=="smilts_aleirits_mals"~"Silt",
                            Litologija=="smilts_aleirits_sapropelis"~"Silt",
                            Litologija=="smilts_dunjaina"~"Sand",
                            Litologija=="smilts_grants"~"Sand",
                            Litologija=="smilts_grants_oli"~"Sand",
                            Litologija=="smilts_grants_oli_aleirits"~"Sand",
                            Litologija=="smilts_kudraina"~"Sand",
                            Litologija=="smilts_malaina_dazadgraudaina,
    ↪ malsmilts"~"Silt",
                            Litologija=="smilts_videjgraudaina, malsmilts"~"
    ↪ Sand",
                            Litologija=="smilts_videjgraudaina"~"Sand",
                            .default=NA))
# numeric codes
kvartars=kvartars %>%
  mutate(grupas_num=case_when(grupas == "Sand" ~"1",
                              grupas == "Silt" ~"2",
                              grupas == "Clay" ~"3",
                              grupas == "organika" ~"4",
                              .default=NA))

# crs transformation
kvartars_3059=st_transform(kvartars,crs=3059)

# nonmissing classes
kvartars_3059=kvartars_3059 %>%
  filter(!is.na(grupas_num))

# rasterization
apaksaugsnem=rasterize(kvartars_3059,template10,field="grupas_num",fun="max",
                       filename="./RasterGrids_10m/2024/SoilTXT_QuarternaryLV
    ↪ .tif",
                       overwrite=TRUE)
```

```
plot(apaksaugsnem)
```

4.7.5 Organic soils: SILAVA

The distribution of organic soils was modelled by EU LIFE Programme project "Demonstration of climate change mitigation potential of nutrients rich organic soils in Baltic States and Finland" at the scientific institue SILAVA. Results are available from their web service: https://silava.forestradar.com/geoserver/silava

Downloaded file was stored at `Geodata/2024/Soils/OrganicSoils_SILAVA/`.

Even tough the layer covers whole of Latvia, it has visible inconsistencies, particularly stripes. These were drawn manually (as vector polygons) and masked out as a part of preprocessing.

For further soil texture analysis we saved a GeoTIFF file with only presences.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# Template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# Organic Soils SILAVA ----

organika_silava=rast("./Geodata/2024/Soils/OrganicSoils_SILAVA/Silava_
    ↪ OrgSoils.tif")
plot(organika_silava)
# visible stripes

# only 40+ cm deep
organika_silava=ifel(organika_silava==2,1,NA)
organika_silavaLV=project(organika_silava,template10)

# stripes drawn manually, rasterization
silavas_telpai=st_read("./IevadesDati/Augsnes_COMB/KudraugsnuPrognozes_Silava
    ↪ /stripam.gpkg",
                       layer="stripam")
silavas_telpai=st_transform(silavas_telpai,crs=3059)
silavas_telpai$yes=1
SilavasTelpa_10=rasterize(silavas_telpai,template10,field="yes")

# presence-only layer without stripes
silava_BezStripam1=ifel(organika_silavaLV==1&SilavasTelpa_10==1,1,NA)
silava_BezStripam=mask(silava_BezStripam1,template10)
plot(silava_BezStripam)
```

```
writeRaster(silava_BezStripam,
            "./RasterGrids_10m/2024/SoilTXT_OrganicSilava.tif",
            overwrite=TRUE)
```

4.7.6 Organic soils: LU

The distribution of organic soils in farmlands was modelled by the University of Latvia project “Improvement of sustainable soil resource management in agriculture”.

From all the results we used layer `YN_prognozes_smooth.tif` stored at `Geodata/2024/Soils/OrganicSoils_LU/`.

Preprocessing consisted of projecting the layer to match 10 m template. Both presences and absences were saved for further processing.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# Template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# Organic Soils LU ----


kudra_norvegi=rast("./Geodata/2024/Soils/OrganicSoils_LU/YN_prognozes_smooth.
    ↪ tif")
kudra_norvLV=project(kudra_norvegi,template10)
plot(kudra_norvLV)

writeRaster(kudra_norvLV,
            "./RasterGrids_10m/2024/SoilTXT_OrganicLU.tif",
            overwrite=TRUE)
```

4.8 Dynamic World data

Dynamic World (DW) is a relatively new Earth observation system product that classifies land cover and land use (LULC) into nine categories (0=water, 1=trees, 2=grass, 3=flooded_vegetation, 4=crops, 5=shrub_and_scrub, 6=built, 7=bare, 8=snow_and_ice), for each ESA Copernicus Sentinel-2 image with identified cloudiness $\leq$35, allowing for filtering and various aggregations (Brown et al., 2022).

DW input information - raster layer for each season in each year - prepared on the Google Earth Engine platform (Gorelick et al., 2017) using a replication script. To use this script, you need a GEE account and project and sufficient space on Google Drive. When executing the command line, a download will be offered for a file covering the time period from the value in row 7 to the value in row 8 (the file name should be specified in row 32, its description in row 33 and the directory on Google Drive in row 31, or all of this can be specified by confirming the save). This script is not optimized for preparing all seasonal sections for all years, so in order to reproduce or expand this study, it is necessary to change it manually.

Downloaded files are to be stored at `Geodata/2024/DynamicWorld/RAW/`.

During download, it can be seen that each layer covering the whole of Latvia is divided into several sheets. This is because, in order to ensure a true zero class (class "water" rather than background), the layers are encoded as Float rather than integers. All of these tiles need to be downloaded, and the following R command lines combine them, ensuring that the coordinate system and pixels correspond to the reference raster.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# 10 m template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# DW export no GEE ----
faili=data.frame(faili=list.files("./Geodata/2024/DynamicWorld/RAW/"))
faili$celi_sakums=paste0("./Geodata/2024/DynamicWorld/RAW/",faili$faili)

# šSagatavoana ----
faili=faili %>%
  separate(faili,into=c("DW","gads","periods","parejais"),sep="_",remove =
    ↪ FALSE) %>%
  mutate(unikalais=paste0(DW,"_",gads,"_",periods),
         mosaic_name=paste0(unikalais,".tif"),
         masaic_cels=paste0("./Geodata/2024/DynamicWorld/",mosaic_name))

# every layer consists of two tiles
unikalie=levels(factor(faili$unikalais))
min(table(faili$unikalais))
max(table(faili$unikalais))

# job
for(i in seq_along(unikalie)){
  unikalais=faili %>% filter(unikalais==unikalie[i])
  beigu_cels=unique(unikalais$masaic_cels)

  print(i)

  viens=rast(unikalais$celi_sakums[1])
  divi=rast(unikalais$celi_sakums[2])
```

```
  viens2=project(viens,template10)
  divi2=project(divi,template10)

  mozaika=mosaic(viens2,divi2,fun="first")
  maskets=mask(mozaika,template10,
               filename=beigu_cels,
               overwrite=TRUE)

  print(beigu_cels)
}
```

4.9 The Global Forest Watch

The Global Forest Watch (GFW) is a widely known product that describes tree canopy cover in 2000, its annual growth from 2001 to 2012, and its annual loss from 2001 to the current version, which is updated annually (Hansen et al., 2013). The data is available both on the project website and on GEE, where it was developed. This project uses v1.12, in which the last year of tree loss dating is 2024, preparing it for download on the GEE platform with this replication script. To use this script, you need a GEE account and project and sufficient space on Google Drive. When executing the command lines, you will be offered to download the file, which you need to save to Google Drive.

After executing the command lines and preparing the results in Google Drive, four files are available for download. The location to download them is `Geodata/2024/Trees/GFW` ↪ `/RAW/`. After download, these files need to be projected to match the reference raster.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}

# 10 m rastra template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# TreeCoverLoss ----
treecoverloss=rast("./Geodata/2024/Trees/GFW/RAW/TreeCoverLoss_v1_12.tif")

tcl=ifel(treecoverloss<1,NA,treecoverloss)

tcl2=terra::project(tcl,paraugs)
tcl3=mask(tcl2,paraugs,filename="./Geodata/2024/Trees/GFW/TreeCoverLoss_v1_
    ↪ 12.tif",overwrite=TRUE)
```

4.10 Palsar

The Palsar Forests resource is based on PALSAR-2 synthetic aperture radar (SAR) reflectance classification of forest and non-forest land with a pixel resolution of 25 m. Forests are classified as areas of at least 0.5 ha covered with trees, where tree cover (at least 5 m high) is at least 10% (Shimada et al., 2013). The data is available at GEE. This project uses a 4-class version (1=Dense Forest, 2=Non-dense Forest, 3=Non-Forest, 4=Water), in which the last tree cover dating year is 2020, prepared for download on the GEE platform with this replication script. To use this script, you need a GEE account and project and sufficient space on Google Drive. When executing the command lines, you will be offered to download the file, which you need to save to Google Drive.

After executing the command lines and preparing the results in Google Drive, four files are available for download. The location to download them is `Geodata/2024/Trees/Palsar/RAW/`. After download, these files need to be projected to match the reference raster and merged. In this resource, trees are coded into two groups: 1=Dense Forest and 2=Non-dense Forest, which need to be merged and the rest converted to missing values (code below).

Although the data in this resource describes the situation in 2020 rather than 2024, it has been used because The Global Forest Watch data is available to describe the disappearance of tree canopy cover, but the appearance of canopy cover is not so rapid that there would be significant changes over a four-year period.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}

# 10 m rastra template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")


# PALSAR Forests ----

fnf1=rast("./Geodata/2024/Trees/Palsar/RAW/ForestNonForest
    ↪ -0000023296-0000023296.tif")
fnf2=rast("./Geodata/2024/Trees/Palsar/RAW/ForestNonForest
    ↪ -0000023296-0000000000.tif")
fnf3=rast("./Geodata/2024/Trees/Palsar/RAW/ForestNonForest
    ↪ -0000000000-0000023296.tif")
fnf4=rast("./Geodata/2024/Trees/Palsar/RAW/ForestNonForest
    ↪ -0000000000-0000000000.tif")

fnf1p=terra::project(fnf1,template10)
fnf2p=terra::project(fnf2,template10)
fnf3p=terra::project(fnf3,template10)
fnf4p=terra::project(fnf4,template10)

fnfA=terra::merge(fnf1p,fnf2p)
fnfB=terra::merge(fnfA,fnf3p)
```

```
fnfC=terra::merge(fnfB,fnf4p)
plot(fnfC)

fnf_X=ifel(fnfC<=2&fnfC>=1,1,NA)
plot(fnf_X)

fnf_XX=mask(fnf_X,template10,
            filename="./Geodata/2024/Trees/Palsar/Palsar_Forests.tif",
            overwrite=TRUE)
```

4.11 CHELSA v2.1

Climatologies at high resolution for the Earth's land surface areas (CHELSA) is 30 arc second global downscaled climate data set (Karger et al., 2017). The temperature algorithm is based on statistical downscaling of atmospheric temperatures. The precipitation algorithm incorporates orographic predictors including wind fields, valley exposition, and boundary layer height, with a subsequent bias correction. CHELSA climatological data has a similar accuracy as other products for temperature, but that its predictions of precipitation patterns are better (Karger et al., 2017). Data (1980-2010 baseline) are freely available for download from homepage forwarding to download server, providing download links for selected products. There is also technical specification available, to decode layer names (https://chelsa-climate.org/wp-admin/download-page/CHELSA_tech_specification_V2.pdf).

The download links we used together with renaming scheme are available with this document. The following command lines perform download, crop to the extent of Latvia (using 1 km vector grid) and saves files for further processing described with other EGVs.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(curl)) {install.packages("curl"); require(curl)}

# templates ----
# 1km grid
tikls1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
telpai=tikls1km %>%
  mutate(yes=1) %>%
  summarise(yes=max(yes)) %>%
  st_buffer(.,dist=10000)

# download and crop ----
```

```
links_names=read_csv("./Geodata/2024/CHELSA/CHELSAdownload_rename.csv")
links_names=links_names %>%
  filter(todownload==1)

for(i in seq_along(links_names$localname)){
  print(i)
  sakums=Sys.time()
  links=links_names$weblocation[i]
  saving1="./Geodata/2024/CHELSA/draza.tif"
  saving2=paste0("./Geodata/2024/CHELSA/",links_names$localname[i])

  curl_download(url=links,destfile = saving1,quiet = FALSE)
  fails=rast(saving1)
  telpa=st_transform(telpai,crs=st_crs(fails))
  nogriezts=crop(fails,telpa,
                 filename=saving2,
                 overwrite=TRUE)
  unlink(saving1)
  beigas=Sys.time()
  ilgums=beigas-sakums
  print(ilgums)
}
```

4.12 HydroClim data

HydroClim is a near-global freshwater-specific environmental variable dataset, created for biodiversity analysis at 1 km resolution (Domisch et al., 2015). Dataset contains many different variables along the HydroSHEDS river network (Lehner et al., 2008), including upstream climate recalculated from worldclim (Hijmans et al., 2005). We downloaded (to `Geodata/2024/HydroClim/`) averaged upstream climate from Zenodo repository (available also from Dryad) and cropped to the extent of Latvia and renamed files for further processing with the code below. Renaming scheme is published with document

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}

# templates ----

template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")
tikls1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
```

```
# reading HydroClim ----
videjie=terra::rast("./Geodata/2024/HydroClim/hydroclim_average+sum.nc")

# reading dictionary -----
slanu_nosaukumi=read_csv("./Geodata/2024/HydroClim/HydroClim_renaming.csv")

# cropping ---
tikls1km_reproj=st_transform(tikls1km,crs=st_crs(videjie))
telpai=tikls1km %>%
  mutate(yes=1) %>%
  summarise(yes=max(yes)) %>%
  st_buffer(.,dist=10000) %>%
  st_transform(.,crs=st_crs(videjie))
videjie=terra::crop(videjie,telpai)

# layer names ----
names(videjie)=slanu_nosaukumi$local_name

# saving files ----
for(i in seq_along(slanu_nosaukumi$local_name)){
  nosaukumam=slanu_nosaukumi$local_name[i]
  writeRaster(videjie[[i]],
              paste0("./Geodata/2024/HydroClim/",nosaukumam),
              overwrite=TRUE)
}
```

The raster dataset contains values only where large enough rivers are detected in HydroSHEDS. However, for species distribution modelling in this project we need continuously covered raster surfaces. For necessary geoprocessing to create such surfaces, we downloaded also HydroBASINS (Lehner and Grill, 2013) dataset to `Geodata/2024/HydroClim/`. These procedures were EGV-specific and are described with other EGVs.

4.13 Sentinel-2 indices

The European Space Agency (ESA) Copernicus program's Sentinel-2 mission is a constellation of two (three since 09/05/2024) identical satellites orbiting in the same orbit. The first satellite, Sentinel-2A, entered its orbit and underwent calibration tests on 2015-06-23, the second (Sentinel-2B) on 2017-03-07, with the first images available earlier. Each satellite captures high-resolution images (from 10 m (at the equator) pixel resolution) in 13 spectral channels with a return time of up to 5 days (more frequently closer to the poles) (https://www.esa.int/Applications/Observing_the_Earth/Copernicus/Sentinel-2). The data from this mission is freely available, including on the Google Earth Engine platform (Gorelick et al., 2017) for various large-scale pre-processing and analysis. We use the harmonized Level-2A (https://developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S2_SR_HARMONIZED#description) product, applying a cloud mask that includes not only cloud filtering but also shadow filtering, so that for each filtered (cloud and seasonal - from April to October and from 2020 to 2024) to calculate the normalized difference vegetation index (NDVI), the normalized difference moisture index (NDMI), and the normalized difference water index (NDWI) as well as various metrics. A replication script can be used to prepare the data. To use this script, you need a GEE account and project and sufficient space on Google Drive. When executing the command lines, the following files will be offered for download:

- `NDVI_median-ST-[runtag, 20250820 by default]` - NDVI short-term median (2020-2024) of annual medians (April to October)
- `NDVI_p25-ST-[runtag, 20250820 by default]` - NDVI short-term median (2020-2024) of annual 25th percentiles (April to October)
- `NDVI_p75-ST-[runtag, 20250820 by default]`- NDVI short-term median (2020-2024) of annual 75th percentiles (April to October)
- `NDVI_iqr-ST-[runtag, 20250820 by default]` - NDVI short-term median (2020-2024) of inter-quartile ranges (April to October)
- `NDVI_median-LY-[runtag, 20250820 by default]` - NDVI last-years (2024) median (April to October)
- `NDMI_median-ST-[runtag, 20250820 by default]` - NDMI short-term median (2020-2024) of annual medians (April to October)
- `NDMI_p25-ST-[runtag, 20250820 by default]` - NDMI short-term median (2020-2024) of annual 25th percentiles (April to October)
- `NDMI_p75-ST-[runtag, 20250820 by default]` - NDMI short-term median (2020-2024) of annual 75th percentiles (April to October)
- `NDMI_iqr-ST-[runtag, 20250820 by default]` - NDMI short-term median (2020-2024) of inter-quartile ranges (April to October)
- `NDMI_median-LY-[runtag, 20250820 by default]` - NDMI last-years (2024) median (April to October)
- `NDWI_median-ST-[runtag, 20250820 by default]` - NDMI short-term median (2020-2024) of annual medians (April to October)
- `NDWI_p25-ST-[runtag, 20250820 by default]` - NDWI short-term median (2020-2024) of annual 25th percentiles (April to October)
- `NDWI_p75-ST-[runtag, 20250820 by default]` - NDWI short-term median (2020-2024) of annual 75th percentiles (April to October)

- `NDWI_iqr-ST-[runtag, 20250820 by default]` - NDWI short-term median (2020-2024) of inter-quartile ranges (April to October)
- `NDWI_median-LY-[runtag, 20250820 by default]` - NDWI last-years (2024) median (April to October)

After executing the command line and preparing the results in Google Drive, it can be seen that each layer covering the whole of Latvia is divided into several tiles. This is because the layers are encoded as Float and exceed 4 GB in size before GeoTIFF compression. All of these files need to be downloaded and located at `Geodata/2024/` ↪ `S2indices/RAW`. The following R commands combine them, ensuring the coordinate systems and its naming, and pixels match the reference raster, while renaming files to `EO_[index]-[term: ST or LY][statistic]`.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}


# 10 m raster template ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# Fails as exported from GEE ----
faili=data.frame(fails=list.files("./Geodata/2024/S2indices/RAW/",pattern = "
    ↪ .tif"))
faili$celi_sakums=paste0("./Geodata/2024/S2indices/RAW/",faili$fails)


# file names ----
faili=faili %>%
  separate(fails,into=c("nosaukums","vidus","beigas"),sep="-",remove = FALSE)
    ↪  %>%
  mutate(mosaic_name=paste0("EO_",nosaukums,"-",beigas,tolower(vidus),".tif")
    ↪ ,
         masaic_cels=paste0("./Geodata/2024/S2indices/Mosaics/",mosaic_name))


unikalie=levels(factor(faili$mosaic_name))
min(table(faili$mosaic_name))
max(table(faili$mosaic_name))

# preparation of mosaics ----
for(i in seq_along(unikalie)){
  sakums=Sys.time()
  unikalais=faili %>% filter(mosaic_name==unikalie[i])
  beigu_cels=unique(unikalais$masaic_cels)

  print(i)
```

```
  # there are exactly 2 tiles per file
  viens=rast(unikalais$celi_sakums[1])
  divi=rast(unikalais$celi_sakums[2])

  viens2=terra::project(viens,template10)
  divi2=terra::project(divi,template10)

  mozaika=terra::merge(viens2,divi2)
  maskets=mask(mozaika,template10,
               filename=beigu_cels,overwrite=TRUE,
               gdal=c("COMPRESS=LZW","TILED=YES","BIGTIFF=IF_SAFER"),
               datatype="FLT4S",
               NAflag=NA)

  plot(maskets,main=unikalie[i])
  print(beigu_cels)
  beigas=Sys.time()
  ilgums=beigas-sakums
  print(ilgums)
}
```

4.14 Waste and garbage disposal sites, landfills

Information on landfills has been compiled from VARAM and Latvian Environment, Geology and Meteorology Center "Report on landfills in Latvia in 2023" listed landfills and their addresses. The coordinates required for the preparation of EGVs were found by combining the resources https://www.google.com/maps and https://balticmaps.eu/. In addition to the resources mentioned above, an object was added at the address "Dardedzes C, Mārupes pag., Mārupes nov., Latvia, LV-2166".

In addition, information from the State Environmental Service on separated waste and deposit packaging collection points was used, exporting it to an Excel file.

Both data sets were combined into a single file and added to this material.

4.15 Digital elevation/terrain models

With the publication of continuous aerial laser scanning data for the territory of Latvia (https://www.lgia.gov.lv/lv/digitalie-augstuma-modeli-0), various high-resolution (1 m and higher) digital surface models (DSM) and digital elevation models (DEM) have been developed. Since the input data is the same in all cases, the values of these (corresponding) models are identical across almost the entire territory of the country. However, airborne laser scanning data (1) is not available for the entire territory of the country, and (2) there are differences between the models in terms of filling (availability of values) outside inland waters and (3) filling of water bodies themselves. However, for areas covered by data on land, the values are almost identical (Pearson's correlation coefficients between the DEMs developed by LU ĢZZF, LVMI Silava, and LĢIA are greater than 0.999999).

The arithmetic mean between the DEMs developed by LU ĢZZF and LVMI Silava, prepared in the University of Latvia project "Improvement of sustainable soil resource management in agriculture", was used as the base DEM. The resolution of this DEM is 1 m, which is not necessary for species distribution modeling input data, therefore the layer is designed to correspond to the reference 10 m raster.

When comparing the projected DEM with the reference, there are clearly distinguishable areas where there is no data. This has been solved by using the DEM with a resolution of 10 m developed by Māris Nartišs (LU ĢZZF) in 2018, which covers the entire territory of Latvia without gaps. To prevent sharp edges from forming in the fill areas (smooth transitions), an arithmetic mean layer was created, covering the entire territory of Latvia and matching the reference raster.

A slope layer has also been created from this raster, which is designed in accordance with the reference. The slope is expressed in degrees and calculated using the 8-neighbor approach. The same applies to the aspect or slope direction.

```
# libs
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}

# reference
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# LiDAR DEM 1 m to 10 m

lapas_1m=data.frame(faili=list.files("./Geodata/2024/DEM/meanDEM_1mOLD/",
    ↪ pattern="*.tif$"))
lapas_1m$numurs=substr(lapas_1m$faili,10,13)
lapas_1m$cels1=paste0("./Geodata/2024/DEM/meanDEM_1mOLD/",lapas_1m$faili)
lapas_1m$cels2=paste0("./Geodata/2024/DEM/meanDEM_10mOLD/",lapas_1m$faili)

kvadrati=st_read(dsn="GIS_Latvija10.2.gdb",layer="tks93_50000")
kvadrati$name=as.character(kvadrati$num50tk)

moz2=rast("./Geodata/2024/DEM/Nartiss_visa_Latvija/dem10_20_kopa.tif")

for(i in 1:length(kvadrati$name)){
  kvadrats=kvadrati[i,]
  nosaukums=kvadrats$name
  telpa=terra::ext(kvadrats)

  paraugs=crop(template10,telpa)
  nart=crop(moz2,telpa)
  nart2=project(nart,paraugs,mask=TRUE)
```

```
  dem1m=lapas_1m[lapas_1m$numurs==kvadrats$name,]
  if(nrow(dem1m)>0){
    sakumcels=dem1m$cels1
    dem=rast(sakumcels)
    reproj=project(dem,paraugs,mask=TRUE,method="bilinear",use_gdal=TRUE)
    videjais <- ifel(is.na(nart2),nart2,ifel(is.na(reproj),nart2,
                                         app(c(nart2,reproj), mean)))
    writeRaster(videjais,overwrite=TRUE,
                filename=paste0("./Geodata/2024/DEM/meanDEM_10m/","vidDEM_",
                                nosaukums,".tif"))
  }
  else{
    writeRaster(nart2,overwrite=TRUE,
                filename=paste0("./Geodata/2024/DEM/meanDEM_10m/","vidDEM_",
                                nosaukums,".tif"))
  }
}

# vrt un mozaic
lapas_10=data.frame(faili=list.files("./Geodata/2024/DEM/meanDEM_10m/",
    ↪ pattern="*.tif$"))
lapas_10$celi1=paste0("./Geodata/2024/DEM/meanDEM_10m/",lapas_10$faili)
mozaikai=vrt(lapas_10$celi1,overwrite=TRUE,
             filename="./Geodata/2024/DEM/vrtDEM_10m.tif")
mozaika=rast("./Geodata/2024/DEM/vrtDEM_10m.tif")
writeRaster(mozaika,"./Geodata/2024/DEM/mozDEM_10m.tif")


## slope
reljefs=rast("./Geodata/2024/DEM/mozDEM_10m.tif")
slipumi=terrain(reljefs, v="slope", neighbors=8, unit="degrees",
                filename="./Geodata/2024/DEM/Terrain_Slope_10m.tif",
    ↪ overwrite=TRUE)


## aspect
reljefs=rast("./Geodata/2024/DEM/mozDEM_10m.tif")
virzieni=terrain(reljefs, v="aspect", neighbors=8, unit="degrees",
                 filename="./Geodata/2024/DEM/Terrain_Aspect_10m.tif",
    ↪ overwrite=TRUE)
```

4.16 Latvian Exclusive Economic Zone polygon

The waters of Latvia's Exclusive Economic Zone were obtained from the HELCOM map and data service. After downloading, this line file was analogically connected to the coastline file obtained from the same resource.

4.17 Bogs and Mires: EDI

Data (training and classification) used in project “Remote Sensing and Machine Learning for Peatland Habitat Monitoring (PurvEO)” by Institute of electronics and computer science are stored at `Geodata/2024/Bogs_EDI`.

Preprocessing combines this information to create two layers:

- `EDI_BogsYN.tif`: training and classification results on open raised bogs (EU protected habitat codes 7110 and 7120) and locations where on of those overlaps with transitional mires (EU protected habitat code 7140);
- `EDI_TransitionalMiresYN.tif`: training and classification results on transitional mires (EU protected habitat code 7140) with no overlap with open rised bogs.

```
# libs
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(terra)) {install.packages("terra"); require(terra)}


# Templates ----

template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")

nulles10=rast("./Templates/TemplateRasters/nulls_LV10m_10km.tif")


# Bogs ----


neatklata71107120=rast("./Geodata/2024/Bogs_EDI/purvi_EDI_projekts/purvi/!LV_
    ↪ kopa_apv1020_30_05_2022/!LV_kopa_apv1020_30_05_2022/Neatklata_purviem_
    ↪ raksturiga_zemsedze_7110_7120.tif")
neatklata71107120=ifel(neatklata71107120>0,1,NA)
plot(neatklata71107120)
neatklata7140=rast("./Geodata/2024/Bogs_EDI/purvi_EDI_projekts/purvi/!LV_kopa
    ↪ _apv1020_30_05_2022/!LV_kopa_apv1020_30_05_2022/Neatklata_purviem_
    ↪ raksturiga_zemsedze_7140.tif")
neatklata7140=ifel(neatklata7140>0,1,NA)

raskturiga71107120=rast("./Geodata/2024/Bogs_EDI/purvi_EDI_projekts/purvi/!LV
    ↪ _kopa_apv1020_30_05_2022/!LV_kopa_apv1020_30_05_2022/Purviem_
    ↪ neraksturiga_zemsedze_7110_7120.tif")
raskturiga71107120=ifel(raskturiga71107120>0,1,NA)
raksturiga7140=rast("./Geodata/2024/Bogs_EDI/purvi_EDI_projekts/purvi/!LV_
    ↪ kopa_apv1020_30_05_2022/!LV_kopa_apv1020_30_05_2022/Purviem_
    ↪ neraksturiga_zemsedze_7140.tif")
```

```
raksturiga7140=ifel(raksturiga7140>0,1,NA)

labels71107120=rast("./Geodata/2024/Bogs_EDI/purvi_EDI_projekts/purvi/!LV_
    ↪ kopa_apv1020_30_05_2022/!LV_kopa_apv1020_30_05_2022/latvija_Labels_
    ↪ B7110_7120.tif")
labels71107120=ifel(labels71107120>0,1,NA)
labels7140=rast("./Geodata/2024/Bogs_EDI/purvi_EDI_projekts/purvi/!LV_kopa_
    ↪ apv1020_30_05_2022/!LV_kopa_apv1020_30_05_2022/latvija_Labels_B7140.
    ↪ tif")
labels7140=ifel(labels7140>0,1,NA)

augstie=cover(cover(neatklata71107120,raksturiga71107120),labels71107120)
parejas=cover(cover(neatklata7140,raksturiga7140),labels7140)
tikai_parejas=ifel(parejas==1&augstie==1,NA,parejas)
sunainie=ifel(parejas==1&augstie==1,parejas,NA)

sunu_purvi=cover(augstie,sunainie)

sunu_proj=project(sunu_purvi,template10)
sunuYN=cover(sunu_proj,nulles10)
plot(sunuYN)
writeRaster(sunuYN,
            overwrite=TRUE,
            filename="./RasterGrids_10m/2024/EDI_BogsYN.tif")



# Transitional mires ----

parejas_proj=project(tikai_parejas,template10)
parejasYN=cover(parejas_proj,nulles10)
plot(parejasYN)
writeRaster(parejasYN,
            overwrite=TRUE,
            filename="./RasterGrids_10m/2024/EDI_TransitionalMiresYN.tif")
```

Chapter 5

Geodata products

Some raw data need extensive processing before EGVs can be created, many EGVs depend on processing raw geodata into geodata products before EGVs can be create, and in some cases, EGV itself could be created from ras geodata, but it has to be spatially restricted to certain locations. This chapter describes these geodata products and procedures involved in creating them.

5.1 Terrain products

In order to develop part of the relief-related EGV, such as the topographic moisture index (TWI) and non-drainage depressions, it is necessary to address water flow in the environment. This is a multi-step procedure that is logical and reliable in mountainous areas and environments with little hydrological impact. However, in the context of Latvia, this is challenging. These challenges can be addressed in various ways. For example, if reliable (accurate) information on the exact locations of rivers and ditches were available, it could be incorporated into the terrain. Unfortunately, there is no sufficiently accurate information available. Therefore, information about network structures from the Melioration Cadastre Information System database buffered by 10 m, bridges from the topographic map and transport structures and bridges from LVM Open Data was used to address the challenges (both buffered by 10 m) - information about the minimum height above sea level was incorporated into the DEM to be used in further processing.

```
# libs
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
```

```
if(!require(exactextractr)){install.packages("exactextractr");require(
    ↪ exactextractr)}

# reference
template=rast("./Templates/TemplateRasters/LV10m_10km.tif")

# reljefs
reljefs=rast("./Geodata/2024/DEM/mozDEM_10m.tif")

# drenazas tikla buves
st_layers("./Geodata/2024/MKIS/MKIS_2025.gpkg")

dtb=st_read("./Geodata/2024/MKIS/MKIS_2025.gpkg",layer="DrenazasTiklaBuves")
dtb_buffer=st_buffer(dtb,dist=10)

# tilti
tiltiL=sfarrow::st_read_parquet("./Geodata/2024/TopographicMap/BridgeL_COMB.
    ↪ parquet")
tiltiL_buffer=st_buffer(tiltiL,dist=30)
tiltiP=sfarrow::st_read_parquet("./Geodata/2024/TopographicMap/BridgeL_COMB.
    ↪ parquet")
tiltiP_buffer=st_buffer(tiltiP,dist=30)

# LVM
lvm_caurtekas=st_read("./Geodata/2024/LVM_OpenData/LVM_CAURTEKAS/LVM_
    ↪ CAURTEKAS_Shape.shp")
lvm_buffer=st_buffer(lvm_caurtekas,dist=30)


# visi buferi
st_geometry(dtb_buffer)="geometry"
st_geometry(tiltiL_buffer)="geometry"
st_geometry(tiltiP_buffer)="geometry"
st_geometry(lvm_buffer)="geometry"
visi_buferi=bind_rows(dtb_buffer,tiltiL_buffer,tiltiP_buffer,lvm_buffer)

# caurumosana
visi_buferi$vertiba=exactextractr::exact_extract(reljefs,visi_buferi,"min")

templis=raster::raster(template)
caurumi=fasterize::fasterize(visi_buferi,templis,field="vertiba")
caurumi2=rast(caurumi)
caurumains=app(c(reljefs,caurumi2),fun="min",na.rm=TRUE,
               overwrite=TRUE,
               filename="./IevadesDati/reljefs/caurtDEM_10m.tif")
```

This DEM was then used for geoprocessing to find terrain depressions and determine the topographic wetness index (TWI). The topographic wetness index was prepared and the search for non-runoff depressions was conducted:

1. drainage depressions and their depth layers were prepared after incorporating flow breaks;
2. to calculate the topographic wetness index, the terrain depressions without runoff were reviewed, allowing up to ten cell breaks in places of lower resistance, the rest were filled in;
3. for additional security, the result of the second step was repeated to search for and fill in terrain depressions (Wang and Liu, 2006);
4. the result of the third step was used to determine the specific catchment area using d-infinity flow division;
5. by combining the specific catchment area layer with the slope layer, the topographic wetness index was calculated. A graphical evaluation revealed individual extreme values, which were limited to **20**.

```r
# libs
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(whitebox)){install.packages("whitebox");require(whitebox)}

# reference
template=rast("./Templates/TemplateRasters/LV10m_10km.tif")


# DEM
caurumainis=rast("./Geodata/2024/DEM/caurtDEM_10m.tif")

# Sinks

## breached sinks un depth in sinks
wbt_breach_depressions_least_cost(
  dem = "./Geodata/2024/DEM/caurtDEM_10m.tif",
  output = "./Geodata/2024/DEM/caurtDEM_breachedNF.tif",
  dist = 10,
  fill = FALSE)
wbt_depth_in_sink(dem="./Geodata/2024/DEM/caurtDEM_breachedNF.tif",
                  output="./Geodata/2024/DEM/Terrain_DiS_breached_10m.tif",
                  zero_background = TRUE)
wbt_sink(input = "./Geodata/2024/DEM/caurtDEM_breachedNF.tif",
         output = "./Geodata/2024/DEM/Terrain_Sink_breached_10m.tif",
         verbose_mode = FALSE,zero_background = TRUE)
sinks=rast("./Geodata/2024/DEM/Terrain_Sink_breached_10m.tif")

sinks2 <- ifel(sinks >= 1, 1, sinks,
               filename="./Geodata/2024/DEM/Terrain_SinkYN_breached_10m.tif")
plot(sinks2)
```

```
# TWI
## breaching
wbt_breach_depressions_least_cost(
  dem = "./Geodata/2024/DEM/caurtDEM_10m.tif",
  output = "./Geodata/2024/DEM/caurtDEM_breachedF.tif",
  dist = 10,
  fill = TRUE)


### filling
wbt_fill_depressions_wang_and_liu(
  dem = "./Geodata/2024/DEM/caurtDEM_breachedF.tif",
  output = "./Geodata/2024/DEM/caurtDEM_BreachFill.tif"
)

### (d inf) flow direction
wbt_d_inf_flow_accumulation(input = "./Geodata/2024/DEM/caurtDEM_BreachFill.
    ↪ tif",
                            output = "./Geodata/2024/DEM/caurtDEM_DInfAccu_
    ↪ SCA.tif",
                            out_type = "Specific Contributing Area")

### twi
wbt_wetness_index(sca = "./Geodata/2024/DEM/caurtDEM_DInfAccu_SCA.tif",
                  slope = "./Geodata/2024/DEM/Terrain_Slope_10m.tif",
                  output = "./Geodata/2024/DEM/TWI_caurtDEM.tif")
twi=rast("./Geodata/2024/DEM/TWI_caurtDEM.tif")
hist(twi) # excessive values
plot(twi)
twi2=ifel(twi>20,20,twi)

writeRaster(twi2x,filename="./Geodata/2024/DEM/Terrain_TWI_lim20_caurtDEM.tif
    ↪ ")
```

Since the initial DEM input was created by filling in water bodies using interpolation methods, the water bodies show a pronounced terrain, which needs to be removed. This is done by inserting average values into these polygons.

```
# libs
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)){install.packages("exactextractr");require(
    ↪ exactextractr)}

# reference
```

```
template=rast("./Templates/TemplateRasters/LV10m_10km.tif")

#  dealing with waterbodies
udeni=sfarrow::st_read_parquet("./Geodata/2024/TopographicMap/HidroA_COMB.
    ↪ parquet")

slope=rast("./Geodata/2024/DEM/Terrain_Slope_10m.tif")
aspect=rast("./Geodata/2024/DEM/Terrain_Aspect_10m.tif")
twi=rast("./Geodata/2024/DEM/Terrain_TWI_lim20_caurtDEM.tif")
dis=rast("./Geodata/2024/DEM/Terrain_DiS_breached_10m.tif")


# average per waterbody
udeni$slopes=exactextractr::exact_extract(slope,udeni,"mean")
caurumi_slope=fasterize::fasterize(udeni,templis,field="slopes")
caurumi_slope2=rast(caurumi_slope)
caurumains_slope=app(c(caurumi_slope2,slope),fun="first",na.rm=TRUE,
                     overwrite=TRUE,
                     filename="./Geodata/2024/DEM/Terrain_Slope_udeni_10m.tif
    ↪ ")
caurumains_slope=terra::rast("./Geodata/2024/DEM/Terrain_Slope_udeni_10m.tif"
    ↪ )
caurumains_slope2=terra::mask(caurumains_slope,template,
                              overwrite=TRUE,
                              filename="./RasterGrids_10m/2024/Terrain_Slope_
    ↪ udeni2_10m.tif")
rm(slope)
rm(caurumi_slope)
rm(caurumi_slope2)
rm(caurumains_slope)
rm(caurumains_slope2)


udeni$aspect=exactextractr::exact_extract(aspect,udeni,"mean")
caurumi_aspect=fasterize::fasterize(udeni,templis,field="aspect")
caurumi_aspect2=rast(caurumi_aspect)
caurumi_aspect=app(c(caurumi_aspect2,aspect),fun="first",na.rm=TRUE,
                   overwrite=TRUE,
                   filename="./Geodata/2024/DEM/Terrain_Aspect_udeni_10m.tif"
    ↪ )
caurumains_aspect=terra::rast("./Geodata/2024/DEM/Terrain_Aspect_udeni_10m.
    ↪ tif")
caurumains_aspect2=terra::mask(caurumains_aspect,template,
                               overwrite=TRUE,
                               filename="./RasterGrids_10m/2024/Terrain_
    ↪ Aspect_udeni2_10m.tif")
rm(aspect)
rm(caurumi_aspect)
rm(caurumi_aspect2)
rm(caurumains_aspect)
```

```
rm(caurumains_aspect2)



udeni$twis=exactextractr::exact_extract(twi,udeni,"mean")
caurumi_TWI=fasterize::fasterize(udeni,templis,field="twis")
caurumi_TWI2=rast(caurumi_TWI)
caurumains_TWI=app(c(caurumi_TWI2,twi),fun="first",na.rm=TRUE,
                   overwrite=TRUE,
                   filename="./Geodata/2024/DEM/Terrain_TWI_udeni_10m.tif")
caurumains_TWI=terra::rast("./Geodata/2024/DEM/Terrain_TWI_udeni_10m.tif")
caurumains_TWI2=terra::mask(caurumains_TWI,template,
                            overwrite=TRUE,
                            filename="./RasterGrids_10m/2024/Terrain_TWI_
    ↪ udeni2_10m.tif")
rm(twi)
rm(caurumi_TWI)
rm(caurumi_TWI2)
rm(caurumains_TWI)
rm(caurumains_TWI2)


udeni$disi=exactextractr::exact_extract(dis,udeni,"mean")
caurumi_DiS=fasterize::fasterize(udeni,templis,field="disi")
caurumi_DiS2=rast(caurumi_DiS)
caurumains_DiS=app(c(caurumi_DiS2,dis),fun="first",na.rm=TRUE,
                   overwrite=TRUE,
                   filename="./Geodata/2024/DEM/Terrain_DiS_udeni_10m.tif")
caurumains_DiS=terra::rast("./Geodata/2024/DEM/Terrain_DiS_udeni_10m.tif")
caurumains_DiS2=terra::mask(caurumains_DiS,template,
                            overwrite=TRUE,
                            filename="./RasterGrids_10m/2024/Terrain_DiS_
    ↪ udeni2_10m.tif")
rm(udeni)
rm(dis)
rm(caurumi_DiS)
rm(caurumi_DiS2)
rm(caurumains_DiS)
rm(caurumains_DiS2)


# cleaning
unlink("./Geodata/2024/DEM/caurtDEM_breachedF.tif")
unlink("./Geodata/2024/DEM/caurtDEM_breachedNF.tif")
unlink("./Geodata/2024/DEM/caurtDEM_BreachFill.tif")
unlink("./Geodata/2024/DEM/caurtDEM_DInfAccu_SCA.tif")

unlink("./Geodata/2024/DEM/Terrain_Slope_udeni_10m.tif")
unlink("./Geodata/2024/DEM/Terrain_Aspect_udeni_10m.tif")
unlink("./Geodata/2024/DEM/Terrain_DiS_udeni_10m.tif")
```

```
unlink("./Geodata/2024/DEM/Terrain_TWI_udeni_10m.tif")
```

5.2 Soil texture product

In this section one united layer describing categorised soil texture (sand=1, silt=2, clay=3, organic=4) is created from multiple preprocessed soil texture data sources. Creation of soil texture product consisted of multiple overlay steps. These steps are illustrated together with processed geodata used:

1. the basis soil texture source was Soil texture from the European Soil Database, this layer had to be reclassified to match other layers as it was not performed during preprocessing;
2. the layer from the first step was overlaid by Latvian Quarternary geology data written as numeric starting with 1;
3. the layer from the second step was overlaid by 20th century topsoil in Latvian farmland written as numeric starting with 1;
4. the layer from Organic soils as modelled by Silava (presence-only) was overlaid by Organic soils as modelled by University of Latvia (presence-absence). After the overlay, it was classified as presence-only;
5. the layer from the third step was overlaid by the layer from the fourth and saved for EGV creation.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}

# step 1
step1=rast("./RasterGrids_10m/2024/SoilTXT_ESDAC.tif")
step1x=ifel(step1==1,1,
            ifel(step1==2,2,
                 ifel(step1==3,2,
                      ifel(step1==4,3,
                           ifel(step1==8,4,NA)))))
plot(step1x)
step1xy=as.numeric(step1x)
plot(step1xy)


# step 2
step2a=rast("./RasterGrids_10m/2024/SoilTXT_QuarternaryLV.tif")
step2a=as.numeric(step2a)+1
plot(step2a)
```

```
step2=cover(step2a,step1x)
plot(step2)

# step 3
step3a=rast("./RasterGrids_10m/2024/SoilTXT_topSoilLV.tif")
step3a=as.numeric(step3a)+1
plot(step3a)

step3=cover(step3a,step2)
plot(step3)

# step 4
step4a=rast("./RasterGrids_10m/2024/SoilTXT_OrganicLU.tif")
step4b=rast("./RasterGrids_10m/2024/SoilTXT_OrganicSilava.tif")

step4c=cover(step4a,step4b)

step4=ifel(step4c==1,4,NA)
plot(step4)

# step 5

step5=cover(step4,step3)
plot(step5)

writeRaster(step5,
           "./RasterGrids_10m/2024/SoilTXT_combined.tif",
           overwrite=TRUE)
```

5.3 Landscape classification

In this exercise, "landscape" refers to the representation of different types of land cover and land use classes, where the order in which these classes are drawn is important, because spatial data from different sources often have mismatched boundaries, which requires addressing both their overlap (1) and filling in gaps where there is no database information (2), and the choice of how to emphasize objects with certain processing, such as buffering, because some elements that are important for characterizing the environment (especially edge effects) may be so small or so poorly positioned that they disappear during the rasterization process (3). The general landscape layer also serves as a mask for the preparation of further environmental descriptions. This section describes the development of a general (simple) landscape and, in the following document, its enrichment with more specific environmental eco-geographical variables. The general landscape is stored in the file `Ainava_vienk_mask.tif`, in which the classes and the procedure for their creation are described in the following list:

- class `100` - **roads**: roads from various sources, **filled in sequence** - dominates classes with higher values so that relatively small objects are not lost and information about edges is provided. The following have been combined to create this class:
  - layers `RoadA_COMB` and `RoadL_COMB` (except smallest size groups) from topographic map, buffered by 10 m before rasterization;
  - LVM open data layers `LVM_MEZA_AUTOCELI`, `LVM_ATTISTAMIE_AUTOCELI`, `LVM_APGRIESANAS_LAUKUMI`, `LVM_IZMAINISANAS_VIETAS` and `LVM_NOBRAUKTUVES` buffered by 10 m;
  - information from the State Forest Register on natural roads has not been used, as these do not usually form a continuous break in the canopy. Information on roads from this register is also available in other resources and has not been duplicated.

The command lines below create a layer with landscape class `100`, which is saved in the file `SimpleLandscape_class100_celi.tif` for further processing.

```
# Libs ----
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}
if(!require(gdalUtilities)){install.packages("gdalUtilities");require(
    gdalUtilities)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)


# class 100 ----

#poly
celi_topo=st_read_parquet("./Geodata/2024/TopographicMap/RoadA_COMB.parquet")
celi_topo=celi_topo %>%
  mutate(yes=100) %>%
  dplyr::select(yes)
ctb=st_buffer(celi_topo,dist=10)
r_celi_topo=fasterize(ctb,template_r,field="yes")

# pts
nobrauktuves=st_read("./Geodata/2024/LVM_OpenData/LVM_NOBRAUKTUVES/LVM_
    NOBRAUKTUVES_Shape.shp")
```

```
nobrauktuves=nobrauktuves %>%
  mutate(yes=100) %>%
  dplyr::select(yes)
izmainisanas=st_read("./Geodata/2024/LVM_OpenData/LVM_IZMAINISANAS_VIETAS/LVM
    ↪ _IZMAINISANAS_VIETAS_Shape.shp")
izmainisanas=izmainisanas %>%
  mutate(yes=100) %>%
  dplyr::select(yes)
apgriesanas=st_read("./Geodata/2024/LVM_OpenData/LVM_APGRIESANAS_LAUKUMI/LVM_
    ↪ APGRIESANAS_LAUKUMI_Shape.shp")
apgriesanas=apgriesanas %>%
  mutate(yes=100) %>%
  dplyr::select(yes)
cp=rbind(nobrauktuves,izmainisanas,apgriesanas)
cpb=st_buffer(cp,dist=10)
r_celi_pts=fasterize(cpb,template_r,field="yes")


# lines
meza_autoceli=st_read("./Geodata/2024/LVM_OpenData/LVM_MEZA_AUTOCELI/LVM_MEZA
    ↪ _AUTOCELI_Shape.shp")
meza_autoceli=meza_autoceli %>%
  mutate(yes=100) %>%
  dplyr::select(yes)
attistamie=st_read("./Geodata/2024/LVM_OpenData/LVM_ATTISTAMIE_AUTOCELI/LVM_
    ↪ ATTISTAMIE_AUTOCELI_Shape.shp")
attistamie=attistamie %>%
  mutate(yes=100) %>%
  dplyr::select(yes)
topo_lines=st_read_parquet("./Geodata/2024/TopographicMap/RoadL_COMB.parquet"
    ↪ )
topo_lines=topo_lines %>%
  mutate(yes=100) %>%
  dplyr::select(yes)
cl=bind_rows(meza_autoceli,attistamie,topo_lines)
cl=cl %>%
  dplyr::select(yes)
clb=st_buffer(cl,dist=10)
r_celi_lines=fasterize(clb,template_r,field="yes")

# cleaning
rm(apgriesanas)
rm(attistamie)
rm(celi_topo)
rm(topo_lines)
rm(ctb)
rm(cl)
rm(clb)
rm(cp)
rm(cpb)
```

```
rm(izmainisanas)
rm(meza_autoceli)
rm(nobrauktuves)

# to terra
t_celi_topo=rast(r_celi_topo)
t_celi_pts=rast(r_celi_pts)
t_celi_lines=rast(r_celi_lines)

# cleaning
rm(r_celi_lines)
rm(r_celi_pts)
rm(r_celi_topo)

# union
plot(t_celi_topo)

road_union1=cover(t_celi_topo,t_celi_pts)
road_union2=cover(road_union1,t_celi_lines,
                  filename="./RasterGrids_10m/2024/SimpleLandscape_class100_
    ↪ celi.tif",
                  overwrite=TRUE)

# cleaning
rm(t_celi_topo)
rm(t_celi_pts)
rm(t_celi_lines)
rm(road_union1)
rm(road_union2)
```

- class `200` - **waters**: water bodies from various sources, filled in sequence (but see "merging and filling" step of this section) - dominates classes with higher values so that relatively small objects are not lost and information about the edges is provided. The following were combined to create this class:
  - topographic map layers `HidroA_COMB` and `HidroL_COMB` (buffered by 5 m);
  - MKIS layer `Gravji`, buffered by 3 m;
  - LVM open data layers `LVM_GRAVJI`, buffered by 5 m.
  - Information about ditches from the State Forest Register has not been used, as it is also available in other resources, or is of so small structures that it does not cause a continuous break in the tree canopy.

The command lines below create a layer with landscape class `200`, which is saved in the file `SimpleLandscape_class200_udens_premask.tif` for further processing.

```
# Libs ----
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
```

```
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}
if(!require(gdalUtilities)){install.packages("gdalUtilities");require(
    ↪ gdalUtilities)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)


# class 200 ----

# topo
topo_udens_poly=st_read_parquet("./Geodata/2024/TopographicMap/HidroA_COMB.
    ↪ parquet")
topo_udens_poly=topo_udens_poly %>%
  mutate(yes=200) %>%
  dplyr::select(yes) %>%
  st_transform(crs=3059)
topo_udens_lines=st_read_parquet("./Geodata/2024/TopographicMap/HidroL_COMB.
    ↪ parquet")
topo_udens_lines=topo_udens_lines %>%
  mutate(yes=200) %>%
  st_buffer(dist=5) %>%
  dplyr::select(yes) %>%
  st_transform(crs=3059)
topo_udens=rbind(topo_udens_poly,topo_udens_lines)
r_topo_udens=fasterize(topo_udens,template_r,field="yes")
raster::writeRaster(r_topo_udens,
                    "./RasterGrids_10m/2024/SimpleLandscape_class200_topo.tif
    ↪ ",
                    progress="text")
# cleaning
rm(topo_udens_lines)
rm(topo_udens_poly)
rm(topo_udens)
rm(r_topo_udens)

# mkis
st_layers("./Geodata/2024/MKIS/MKIS_2025.gpkg")
mkis_gravji=st_read("./Geodata/2024/MKIS/MKIS_2025.gpkg",layer="Gravji")

mkis_gravji=mkis_gravji %>%
  mutate(yes=200) %>%
  st_buffer(dist=3) %>%
```

```
  dplyr::select(yes)
r_mkis_udens=fasterize(mkis_gravji,template_r,field="yes")
raster::writeRaster(r_mkis_udens,
                    "./RasterGrids_10m/2024/SimpleLandscape_class200_mkis.tif
    ↪ ",
                    progress="text")
# cleaning
rm(mkis_gravji)
rm(mkis_gravji2)
rm(mkis_gravji3)
rm(r_mkis_udens)

# lvm
lvm_gravji=st_read("./Geodata/2024/LVM_OpenData/LVM_GRAVJI/LVM_GRAVJI_Shape.
    ↪ shp")
lvm_gravji=lvm_gravji %>%
  mutate(yes=200) %>%
  st_buffer(dist=5) %>%
  dplyr::select(yes)
r_lvm_gravji=fasterize(lvm_gravji,template_r,field="yes")
raster::writeRaster(r_lvm_gravji,
                    "./RasterGrids_10m/2024/SimpleLandscape_class200_lvm.tif"
    ↪ ,
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(lvm_gravji)
rm(r_lvm_gravji)


# merging
a200=rast("./RasterGrids_10m/2024/SimpleLandscape_class200_topo.tif")
b200=rast("./RasterGrids_10m/2024/SimpleLandscape_class200_mkis.tif")
c200=rast("./RasterGrids_10m/2024/SimpleLandscape_class200_lvm.tif")

udens_cover1=cover(a200,b200)
udens_cover2=cover(udens_cover1,c200,
                   filename="./RasterGrids_10m/2024/SimpleLandscape_class200_
    ↪ udens_premask.tif",
                   overwrite=TRUE)

# cleaning
rm(a200)
rm(b200)
rm(c200)
rm(udens_cover1)
rm(udens_cover2)
unlink("./RasterGrids_10m/2024/SimpleLandscape_class200_topo.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class200_mkis.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class200_lvm.tif")
```

- class `300` - **farmland**: agricultural land in LAD database **filled in sequence** - dominated by classes with higher values, but after the general classes were created, the gaps were filled in with information from Dynamic World. The following were combined to create this class:

  – LAD database, which, following the decision on grouping (classes are available here), is divided into three broad groups (in order of overlap):

  – arable land with class code `310`;

  – fallow land with class code `320`;

  – grassland with class code `330`;

  – orchards and perennial shrub plantations in the general landscape are placed in other landscape classes.

The command lines below create a layer with landscape class `300` and its subclasses, which are saved in the file `SimpleLandscape_class300_lauki_premask.tif` for further processing.

```
# Libs ----
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}
if(!require(gdalUtilities)){install.packages("gdalUtilities");require(
    ↪ gdalUtilities)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)


# class 300 ----

# lad
lad_klasem=read_excel("./Geodata/2024/LAD/KulturuKodi_2024.xlsx")
lad=st_read_parquet("./Geodata/2024/LAD/Lauki_2024.parquet")


## arable
amazemem=lad_klasem %>%
  filter(str_detect(SDM_grupa_sakums,"aramz"))
```

```
aramzemes=lad %>%
  filter(PRODUCT_CODE %in% amazemem$kods) %>%
  mutate(yes=310) %>%
  dplyr::select(yes)
r_aramzemes_lad=fasterize(aramzemes,template_r,field="yes")
raster::writeRaster(r_aramzemes_lad,
                    "./RasterGrids_10m/2024/SimpleLandscape_class310_
    ↪ aramzemes_lad.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(amazemem)
rm(aramzemes)
rm(r_aramzemes_lad)


## fallow
papuvem=lad_klasem %>%
  filter(str_detect(SDM_grupa_sakums,"papuv"))
papuves=lad %>%
  filter(PRODUCT_CODE %in% papuvem$kods) %>%
  mutate(yes=320) %>%
  dplyr::select(yes)
r_papuves_lad=fasterize(papuves,template_r,field="yes")
raster::writeRaster(r_papuves_lad,
                    "./RasterGrids_10m/2024/SimpleLandscape_class320_papuves_
    ↪ lad.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(papuvem)
rm(papuves)
rm(r_papuves_lad)

## grassland
zalajiem=lad_klasem %>%
  filter(str_detect(SDM_grupa_sakums,"āāzl"))
zalaji=lad %>%
  filter(PRODUCT_CODE %in% zalajiem$kods) %>%
  mutate(yes=330) %>%
  dplyr::select(yes)
r_zalaji_lad=fasterize(zalaji,template_r,field="yes")
raster::writeRaster(r_zalaji_lad,
                    "./RasterGrids_10m/2024/SimpleLandscape_class330_zalaji_
    ↪ lad.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(zalajiem)
rm(zalaji)
```

```
rm(r_zalaji_lad)

# merging
a300=rast("./RasterGrids_10m/2024/SimpleLandscape_class310_aramzemes_lad.tif"
    ↪ )
b300=rast("./RasterGrids_10m/2024/SimpleLandscape_class320_papuves_lad.tif")
c300=rast("./RasterGrids_10m/2024/SimpleLandscape_class330_zalaji_lad.tif")

farmland_cover1=cover(a300,b300)
farmland_cover2=cover(farmland_cover1,c300,
                          filename="./RasterGrids_10m/2024/SimpleLandscape_
    ↪ class300_lauki_premask.tif",
                          overwrite=TRUE)
# cleaning
rm(lad)
rm(lad_klasem)
rm(a300)
rm(b300)
rm(c300)
rm(farmland_cover1)
rm(farmland_cover2)
unlink("./RasterGrids_10m/2024/SimpleLandscape_class310_aramzemes_lad.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class320_papuves_lad.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class330_zalaji_lad.tif")
```

- class `400` - **allotment gardens and orchards, cottages**, **filled in order** - dominate classes with higher values. To create this class, the following were combined (in order of overlap):
  - topographic map layer `LandusA_COMB`, the result of which is coded with `410`;
  - LAD database rural information layer group (classes are available here) “augļudārzi”, the result of which is coded with `420`.

The command lines below create a layer with landscape class `400`, which is saved in the file `SimpleLandscape_class400_vasarnicas_premask.tif` for further processing.

```
# Libs ----
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
```

```
template_r=raster(template_t)


# class 400 ----

# topo
darzini_topo=st_read_parquet("./Geodata/2024/TopographicMap/LandusA_COMB.
    ↪ parquet")
table(darzini_topo$FNAME,useNA="always")
darzini_topo=darzini_topo %>%
  filter(FNAME %in% c("poligons_Augludarzs","poligons_ļāAugudrzs","poligons_
    ↪ ņāSakudrzs",
                      "poligons_āOguljs","poligons_Ogulajs","poligons_
    ↪ Saknudarzs")) %>%
  mutate(yes=410) %>%
  dplyr::select(yes)
r_darzini_topo=fasterize(darzini_topo,template_r,field="yes")
raster::writeRaster(r_darzini_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class410_darzini_
    ↪ topo.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(darzini_topo)
rm(r_darzini_topo)

# lad
lad_klasem=read_excel("./Geodata/2024/LAD/KulturuKodi_2024.xlsx")
table(lad_klasem$SDM_grupa_sakums,useNA="always")
augludarziem=lad_klasem %>%
  filter(SDM_grupa_sakums=="ļāaugudrzi")
lad=st_read_parquet("./Geodata/2024/LAD/Lauki_2024.parquet")
lad=lad %>%
  filter(PRODUCT_CODE %in% augludarziem$kods) %>%
  mutate(yes=420) %>%
  dplyr::select(yes)
r_darzini_lad=fasterize(lad,template_r,field="yes")
raster::writeRaster(r_darzini_lad,
                    "./RasterGrids_10m/2024/SimpleLandscape_class420_darzini_
    ↪ lad.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(lad_klasem)
rm(augludarziem)
rm(lad)
rm(r_darzini_lad)

# merging
a400=rast("./RasterGrids_10m/2024/SimpleLandscape_class410_darzini_topo.tif")
```

```
b400=rast("./RasterGrids_10m/2024/SimpleLandscape_class420_darzini_lad.tif")

allotment_cover=cover(a400,b400,
                     filename="./RasterGrids_10m/2024/SimpleLandscape_
    ↪ class400_varnicas_premask.tif",
                     overwrite=TRUE)

# cleaning
rm(a400)
rm(b400)
rm(allotment_cover)
unlink("./RasterGrids_10m/2024/SimpleLandscape_class410_darzini_topo.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class420_darzini_lad.tif")
```

- class `500` - **built-up**: built-up areas, filled in at the end (see section "merging and filling" of this chapter) using information from Dynamic World about places not covered by other classes.

- class `600` - **forests, shrublands, clearings**: areas covered with trees and shrubs, clearings, and dead forest stands, **filled in order** - dominates classes with higher values. The following have been combined to create this class (in order of overlap):

  – The Global Forest Watch layer records of tree canopy cover loss since 2020, coded as `610`;

  – Forest State Register clearings and dead forest stands, the result of which is coded as `610`;

  – Forest State Register marked forest stands that are lower than 5 m and seed production plantations, the result of which is coded as `620`;

  – topographic map layer `FloraL_COMB` classes related to shrubs, buffered by 10 m, coded as `620`;

  – topographic map layers `LandusA_COMB` clases "poligons_Krūmājs", "poligons_Krumajs", "poligons_Krūmaugu_plant", "poligons_Plantacija_krum", coded as `620`;

  – LAD database group (classes are available here) "krūmveida ilggadīgie stādījumi", the result of which is coded with `620`;

  – Forest State Register forest stands with a height of at least 5 m, coded as `630`;

  – topographic map layer `LandusA_COMB` classes "poligons_Parks", "poligons_Meza_kapi", "poligons_Kapi", "poligons_Kapi_meza", the result of which is coded as `640`;

  – topographic map layer `FloraL_COMB` with tree-related classes buffered by 10 m, coded as `640`;

  – Palsar Forests layer, coded as `630`.

The command lines below create a layer with landscape class `600`, which is saved in the file `SimpleLandscape_class600_meziem_premask.tif` for further processing.

```r
# Libs ----
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)


# class 600 ----

# mvr
mvr=st_read_parquet("./Geodata/2024/MVR/nogabali_2024janv.parquet")

# clearcuts
izcirtumi=mvr %>%
  filter(zkat %in% c("12","14")) %>%
  mutate(yes=610) %>%
  dplyr::select(yes)
r_izcirtumi_mvr=fasterize(izcirtumi,template_r,field="yes")
raster::writeRaster(r_izcirtumi_mvr,
                    "./RasterGrids_10m/2024/SimpleLandscape_class610_
    ↪ izcirtumi_mvr.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(izcirtumi)
rm(r_izcirtumi_mvr)

# low stands
# also zkat 16
zemas_audzes=mvr %>%
  filter((zkat =="10" & h10<5)|zkat=="16") %>%
  mutate(yes=620) %>%
  dplyr::select(yes)
r_zemas_mvr=fasterize(zemas_audzes,template_r,field="yes")
raster::writeRaster(r_zemas_mvr,
                    "./RasterGrids_10m/2024/SimpleLandscape_class620_zemas_
    ↪ mvr.tif",
                    progress="text",
                    overwrite=TRUE)
```

```
# cleaning
rm(zemas_audzes)
rm(r_zemas_mvr)


# high stands
augstas_audzes=mvr %>%
  filter(zkat =="10" & h10>=5) %>%
  mutate(yes=630) %>%
  dplyr::select(yes)
r_augstas_mvr=fasterize(augstas_audzes,template_r,field="yes")
raster::writeRaster(r_augstas_mvr,
                    "./RasterGrids_10m/2024/SimpleLandscape_class630_augstas_
    ↪ mvr.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(augstas_audzes)
rm(r_augstas_mvr)
rm(mvr)

# tcl - since 2020
tcl=rast("./Geodata/2024/Trees/GFW/TreeCoverLoss_v1_12.tif")
tcl2=ifel(tcl<20,NA,610,
          filename="./RasterGrids_10m/2024/SimpleLandscape_class610_TCL.tif",
          overwrite=TRUE)
# cleaning
rm(tcl)
rm(tcl2)

# palsar
palsar=rast("./Geodata/2024/Trees/Palsar/Palsar_Forests.tif")
palsar2=ifel(palsar==1,630,NA,
             filename="./RasterGrids_10m/2024/SimpleLandscape_class630_Palsar
    ↪ .tif",
             overwrite=TRUE)
# cleaning
rm(palsar)
rm(palsar2)


# lad
lad_klasem=read_excel("./Geodata/2024/LAD/KulturuKodi_2024.xlsx")
table(lad_klasem$SDM_grupa_sakums,useNA="always")
lad=st_read_parquet("./Geodata/2024/LAD/Lauki_2024.parquet")
krumiem=lad_klasem %>%
  filter(str_detect(SDM_grupa_sakums,"ūkrmv"))
krumi=lad %>%
  filter(PRODUCT_CODE %in% krumiem$kods) %>%
  mutate(yes=620) %>%
```

```
  dplyr::select(yes)
r_krumi_lad=fasterize(krumi,template_r,field="yes")
raster::writeRaster(r_krumi_lad,
                    "./RasterGrids_10m/2024/SimpleLandscape_class620_krumi_
    ↪ lad.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(lad_klasem)
rm(lad)
rm(krumiem)
rm(krumi)
rm(r_krumi_lad)

# topo - pkk
pkk_topo=st_read_parquet("./Geodata/2024/TopographicMap/LandusA_COMB.parquet"
    ↪ )
table(pkk_topo$FNAME,useNA="always")
pkk_topo=pkk_topo %>%
  filter(FNAME %in% c("poligons_Parks","poligons_Meza_kapi","poligons_Kapi",
                      "poligons_Kapi_meza")) %>%
  mutate(yes=640) %>%
  dplyr::select(yes)
r_pkk_topo=fasterize(pkk_topo,template_r,field="yes")
raster::writeRaster(r_pkk_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class640_pkk_topo
    ↪ .tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(pkk_topo)
rm(r_pkk_topo)

# topo - shrubs
krumi_topo=st_read_parquet("./Geodata/2024/TopographicMap/LandusA_COMB.
    ↪ parquet")
table(krumi_topo$FNAME,useNA="always")
krumi_topo=krumi_topo %>%
  filter(FNAME %in% c("poligons_ūāKrmjs","poligons_Krumajs",
                      "poligons_ūKrmaugu_plant","poligons_Plantacija_krum"))
    ↪ %>%
  mutate(yes=620) %>%
  dplyr::select(yes)
r_krumi_topo=fasterize(krumi_topo,template_r,field="yes")
raster::writeRaster(r_krumi_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class620_krumi_
    ↪ topo.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
```

```
rm(krumi_topo)
rm(r_krumi_topo)

# topo - linear vegetation
linijas_topo=st_read_parquet("./Geodata/2024/TopographicMap/FloraL_COMB.
    ↪ parquet")
table(linijas_topo$FNAME,useNA="always")

# linear shrubs
krumu_linijas_topo=linijas_topo %>%
  filter(FNAME=="ūKrmu rinda īdzvzogs"|FNAME=="ūKrmu rinda gar ļceiem ēupm"|
           FNAME=="Krumu_rinda_dzivzogs"|FNAME=="Krumu_rinda_gar_celiem_upem"
    ↪ ) %>%
  mutate(yes=620) %>%
  st_buffer(dist=10) %>%
  dplyr::select(yes)
r_krumu_linijas_topo=fasterize(krumu_linijas_topo,template_r,field="yes")
raster::writeRaster(r_krumu_linijas_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class620_
    ↪ KrumuLinijas_topo.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(krumu_linijas_topo)
rm(r_krumu_linijas_topo)

# linear trees
koku_linijas_topo=linijas_topo %>%
  filter(str_detect(FNAME,"Koku")) %>%
  mutate(yes=640) %>%
  st_buffer(dist=10) %>%
  dplyr::select(yes)
r_koku_linijas_topo=fasterize(koku_linijas_topo,template_r,field="yes")
raster::writeRaster(r_koku_linijas_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class640_
    ↪ KokuLinijas_topo.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(koku_linijas_topo)
rm(r_koku_linijas_topo)
rm(linijas_topo)

# merging
r_krumi_lad=rast("./RasterGrids_10m/2024/SimpleLandscape_class620_krumi_lad.
    ↪ tif")
r_pkk_topo=rast("./RasterGrids_10m/2024/SimpleLandscape_class640_pkk_topo.tif
    ↪ ")
r_krumi_topo=rast("./RasterGrids_10m/2024/SimpleLandscape_class620_krumi_topo
    ↪ .tif")
```

```
r_krumu_linijas_topo=rast("./RasterGrids_10m/2024/SimpleLandscape_class620_
    ↪ KrumuLinijas_topo.tif")
r_koku_linijas_topo=rast("./RasterGrids_10m/2024/SimpleLandscape_class640_
    ↪ KokuLinijas_topo.tif")
r_palsar=rast("./RasterGrids_10m/2024/SimpleLandscape_class630_palsar.tif")
r_tcl=rast("./RasterGrids_10m/2024/SimpleLandscape_class610_TCL.tif")
r_augstas_mvr=rast("./RasterGrids_10m/2024/SimpleLandscape_class630_augstas_
    ↪ mvr.tif")
r_zemas_mvr=rast("./RasterGrids_10m/2024/SimpleLandscape_class620_zemas_mvr.
    ↪ tif")
r_izcirtumi_mvr=rast("./RasterGrids_10m/2024/SimpleLandscape_class610_
    ↪ izcirtumi_mvr.tif")


mezu_cover=cover(r_tcl,r_izcirtumi_mvr)
mezu_cover=cover(mezu_cover,r_zemas_mvr)
mezu_cover=cover(mezu_cover,r_krumu_linijas_topo)
mezu_cover=cover(mezu_cover,r_krumi_topo)
mezu_cover=cover(mezu_cover,r_krumi_lad)
mezu_cover=cover(mezu_cover,r_augstas_mvr)
mezu_cover=cover(mezu_cover,r_pkk_topo)
mezu_cover=cover(mezu_cover,r_koku_linijas_topo)
mezu_cover=cover(mezu_cover,r_palsar,
                 filename="./RasterGrids_10m/2024/SimpleLandscape_class600_
    ↪ meziem_premask.tif",
                 overwrite=TRUE)


# cleaning
rm(r_krumi_lad)
rm(r_pkk_topo)
rm(r_krumi_topo)
rm(r_krumu_linijas_topo)
rm(r_koku_linijas_topo)
rm(r_palsar)
rm(r_tcl)
rm(r_augstas_mvr)
rm(r_zemas_mvr)
rm(r_izcirtumi_mvr)
rm(mezu_cover)

unlink("./RasterGrids_10m/2024/SimpleLandscape_class620_krumi_lad.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class640_pkk_topo.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class620_krumi_topo.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class620_KrumuLinijas_topo.tif
    ↪ ")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class640_KokuLinijas_topo.tif"
    ↪ )
unlink("./RasterGrids_10m/2024/SimpleLandscape_class630_palsar.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class610_TCL.tif")
```

```
unlink("./RasterGrids_10m/2024/SimpleLandscape_class630_augstas_mvr.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class620_zemas_mvr.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class610_izcirtumi_mvr.tif")
```

- class `700` - **wetlands**: combining geospatial data related to reed beds, marshes, mires and bogs, **filled in order except class `720` that dominates over waters** - dominates classes with higher values. To create this class, the following were combined (in order of overlap):

  – topographic map layer `LandusA_COMB` classes “Meldrājs_ūdenī_poligons”, “poligons_Grislajs”, “poligons_Grīslājs”, “poligons_Meldrajs”, “poligons_Meldrājs”, “poligons_Meldrajs_udeni”, “poligons_Nec_purvs_grīslājs”, “poligons_Nec_purvs_meldrājs”, “Sēklis_poligons”, the result of which is coded with `720`;

  – topographic map layer `LandusA_COMB` class “poligons_Nec_purvs_sūnājs”, “poligons_Sunajs”, “poligons_Sūnājs”, the result of which is coded with `710`;

  – topographic map layer `SwampA_COMB`, the result of which is coded as `710`;

  – land categories “21”, “22”, “23” marked in the State Forest Register, the result of which is coded as `710`;

  – land categories “41” and “42” marked in the State Forest Register, the result of which is coded as `730`;

  - bogs from Bogs and Mires: EDI;
  - transitional mires from Bogs and Mires: EDI;

The command lines below create a layer with landscape class `700`, which is saved in the file `SimpleLandscape_class700_mitraji_premask.tif` for further processing.

```
# Libs ----
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)


# class 700 ----
```

```
# topo
topo=st_read_parquet("./Geodata/2024/TopographicMap/LandusA_COMB.parquet")
table(topo$FNAME,useNA="always")

## ReedSedgeRush
niedraji_topo=topo %>%
  filter(FNAME %in% c("āMeldrjs_ūīden_poligons","poligons_Grislajs","poligons
    ↪ _īāGrsljs",
                      "poligons_Meldrajs","poligons_āMeldrjs","poligons_
    ↪ Meldrajs_udeni",
                      "poligons_Nec_purvs_īāgrsljs",
                      "poligons_Nec_purvs_āmeldrjs",
                      "ēSklis_poligons")) %>%
  mutate(yes=720) %>%
  dplyr::select(yes)
r_niedraji_topo=fasterize(niedraji_topo,template_r,field="yes")
raster::writeRaster(r_niedraji_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class720_niedraji
    ↪ _topo.tif",
                    progress="text")
# cleaning
rm(niedraji_topo)
rm(r_niedraji_topo)


## bogs
purvi_topo=topo %>%
  filter(FNAME %in% c("poligons_Nec_purvs_ūāsnjs",
                      "poligons_Sunajs","poligons_ūāSnjs")) %>%
  mutate(yes=710) %>%
  dplyr::select(yes)
topo_purvi=st_read_parquet("./Geodata/2024/TopographicMap/SwampA_COMB.parquet
    ↪ ")
topo_purvi=topo_purvi %>%
  mutate(yes=710) %>%
  dplyr::select(yes)
purvi=rbind(purvi_topo,topo_purvi)
r_purvi_topo=fasterize(purvi,template_r,field="yes")
raster::writeRaster(r_purvi_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class710_purvi_
    ↪ topo.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(purvi_topo)
rm(topo_purvi)
rm(purvi)
rm(r_purvi_topo)
```

```
# mvr
mvr=st_read_parquet("./Geodata/2024/MVR/nogabali_2024janv.parquet")

# bogs and mires
mvr_purvi=mvr %>%
  filter(zkat %in% c("21","22","23")) %>%
  mutate(yes=710) %>%
  dplyr::select(yes)
r_purvi_mvr=fasterize(mvr_purvi,template_r,field="yes")
raster::writeRaster(r_purvi_mvr,
                    "./RasterGrids_10m/2024/SimpleLandscape_class710_purvi_
    ↪ mvr.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(mvr_purvi)
rm(r_purvi_mvr)

# beavers
mvr_bebri=mvr %>%
  filter(zkat %in% c("41","42")) %>%
  mutate(yes=730) %>%
  dplyr::select(yes)
r_bebri_mvr=fasterize(mvr_bebri,template_r,field="yes")
raster::writeRaster(r_bebri_mvr,
                    "./RasterGrids_10m/2024/SimpleLandscape_class730_bebri_
    ↪ mvr.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(mvr_bebri)
rm(r_bebri_mvr)
rm(mvr)



# merging
r_niedraji_topo=rast("./RasterGrids_10m/2024/SimpleLandscape_class720_
    ↪ niedraji_topo.tif")
r_purvi_topo=rast("./RasterGrids_10m/2024/SimpleLandscape_class710_purvi_topo
    ↪ .tif")
r_purvi_mvr=rast("./RasterGrids_10m/2024/SimpleLandscape_class710_purvi_mvr.
    ↪ tif")
r_bebri_mvr=rast("./RasterGrids_10m/2024/SimpleLandscape_class730_bebri_mvr.
    ↪ tif")
mires=rast("./RasterGrids_10m/2024/EDI_TransitionalMiresYN.tif")
miresY=ifel(mires==1,710,NA)
bogs=rast("./RasterGrids_10m/2024/EDI_BogsYN.tif")
bogsY=ifel(bogs==1,710,NA)
```

```
wetlands_cover=cover(r_niedraji_topo,r_purvi_topo)
wetlands_cover=cover(wetlands_cover,r_purvi_mvr)
wetlands_cover=cover(wetlands_cover,r_bebri_mvr)
wetlands_cover=cover(wetlands_cover,miresY)
wetlands_cover=cover(wetlands_cover,bogsY,
                              filename="./RasterGrids_10m/2024/SimpleLandscape_
    ↪ class700_mitraji_premask.tif",
                              overwrite=TRUE)
# cleaning
rm(r_niedraji_topo)
rm(r_purvi_topo)
rm(r_purvi_mvr)
rm(r_bebri_mvr)
rm(bogs)
rm(mires)
rm(bogsY)
rm(miresY)
rm(topo)
rm(wetlands_cover)

unlink("./RasterGrids_10m/2024/SimpleLandscape_class710_purvi_topo.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class710_purvi_mvr.tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class730_bebri_mvr.tif")
```

- class `800` - **bare soil and quarries**: combining layers related to bare soil, heaths, and quarries, **filled in order** - as this is the highest class, it dominates only over Dynamic World used to fill gaps. The following have been combined to create this class (in order of overlap):
  - topographic map layer `LandusA_COMB` classes "poligons_Smiltājs", "poligons_Smiltajs", "poligons_Grants", "poligons_Kūdra", "poligons_Virsajs" the result of which is coded with `800`;
  - land categories "33" and "34" marked in the State Forest Register, the result of which is coded as `800`.

The command lines below create a layer with landscape class `800`, which is saved in the file `SimpleLandscape_class800_smiltaji_premask.tif` for further processing.

```
# Libs ----
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}
```

```
# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)


# class 800 ----

smiltaji_topo=st_read_parquet("./Geodata/2024/TopographicMap/LandusA_COMB.
    ↪ parquet")
table(smiltaji_topo$FNAME,useNA="always")
smiltaji_topo=smiltaji_topo %>%
  filter(FNAME %in% c("poligons_āSmiltjs","poligons_Smiltajs","poligons_
    ↪ Grants",
                      "poligons_ūKdra","poligons_Virsajs")) %>%
  mutate(yes=800) %>%
  dplyr::select(yes)
r_smiltaji_topo=fasterize(smiltaji_topo,template_r,field="yes")
raster::writeRaster(r_smiltaji_topo,
                    "./RasterGrids_10m/2024/SimpleLandscape_class800_
    ↪ SmiltajiKudra_topo.tif",
                    progress="text")
# cleaning
rm(smiltaji_topo)
rm(r_smiltaji_topo)

# mvr zkat 33 un 34
mvr=st_read_parquet("./Geodata/2024/MVR/nogabali_2024janv.parquet")

smiltajiem=mvr %>%
  filter(zkat %in% c("33","34")) %>%
  mutate(yes=800) %>%
  dplyr::select(yes)
r_smiltaji_mvr=fasterize(smiltajiem,template_r,field="yes")
raster::writeRaster(r_smiltaji_mvr,
                    "./RasterGrids_10m/2024/SimpleLandscape_class800_
    ↪ SmiltVirs_mvr.tif",
                    progress="text",
                    overwrite=TRUE)
# cleaning
rm(mvr)
rm(smiltajiem)
rm(r_smiltaji_mvr)

# merging
r_smiltaji_topo=rast("./RasterGrids_10m/2024/SimpleLandscape_class800_
    ↪ SmiltajiKudra_topo.tif")
r_smiltaji_mvr=rast("./RasterGrids_10m/2024/SimpleLandscape_class800_
    ↪ SmiltVirs_mvr.tif")

bare_cover=terra::merge(r_smiltaji_topo,r_smiltaji_mvr,
```

```
                              filename="./RasterGrids_10m/2024/
    ↪ SimpleLandscape_class800_smiltaji_premask.tif",
                              overwrite=TRUE)
# āliek āšaizvkana
rm(r_smiltaji_topo)
rm(r_smiltaji_mvr)
rm(bare_cover)

unlink("./RasterGrids_10m/2024/SimpleLandscape_class800_SmiltajiKudra_topo.
    ↪ tif")
unlink("./RasterGrids_10m/2024/SimpleLandscape_class800_SmiltVirs_mvr.tif")
```

Merging and filling

The command lines below combine the previously created layers with landscape classes in the correct order and ensure that gaps are filled with the appropriately classified Dynamic World composite for April-August 2024. After masking to only the analysis space, layer is saved in the in the file `Ainava_vienk_mask.tif` for further processing.

```
# Libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)



# final merging and covering ----

# DW
dynworld=rast("Geodata/2024/DynamicWorld/DW_2024_apraug.tif")
klases=matrix(c(0,200,
                1,620,
                2,330,
                3,720,
                4,310,
                5,710,
                6,500,
                7,800,
                8,500),ncol=2,byrow=TRUE)
dw2=terra::classify(dynworld,klases)
writeRaster(dw2,
            "./RasterGrids_10m/2024/DW_reclass.tif",
            overwrite=TRUE)
# other layers
celi=rast("./RasterGrids_10m/2024/SimpleLandscape_class100_celi.tif")
niedraji=rast("RasterGrids_10m/2024/SimpleLandscape_class720_niedraji_topo.
    ↪ tif")
```

```
udeni=rast("./RasterGrids_10m/2024/SimpleLandscape_class200_udens_premask.tif
    ↪ ")
lauki=rast("./RasterGrids_10m/2024/SimpleLandscape_class300_lauki_premask.tif
    ↪ ")
vasarnicas=rast("./RasterGrids_10m/2024/SimpleLandscape_class400_varnicas_
    ↪ premask.tif")
mezi=rast("./RasterGrids_10m/2024/SimpleLandscape_class600_meziem_premask.tif
    ↪ ")
mitraji=rast("./RasterGrids_10m/2024/SimpleLandscape_class700_mitraji_premask
    ↪ .tif")
smiltaji=rast("./RasterGrids_10m/2024/SimpleLandscape_class800_smiltaji_
    ↪ premask.tif")
dw2=rast("./RasterGrids_10m/2024/DW_reclass.tif")

# covering in correct order
rastri_ainavai=cover(celi,niedraji)
rastri_ainavai=cover(rastri_ainavai,udeni)
rastri_ainavai=cover(rastri_ainavai,lauki)
rastri_ainavai=cover(rastri_ainavai,vasarnicas)
rastri_ainavai=cover(rastri_ainavai,mezi)
rastri_ainavai=cover(rastri_ainavai,mitraji)
rastri_ainavai=cover(rastri_ainavai,smiltaji)
rastri_ainavai=cover(rastri_ainavai,dw2,
                           filename="./RasterGrids_10m/2024/Ainava_vienkarsa.
    ↪ tif",
                           overwrite=TRUE)

# cleaning
rm(celi)
rm(niedraji)
rm(udeni)
rm(lauki)
rm(vasarnicas)
rm(mezi)
rm(mitraji)
rm(smiltaji)
rm(klases)
rm(dynworld)
rm(dw2)
rm(rastri_ainavai)

# masking
rastrs_ainava=rast("./RasterGrids_10m/2024/Ainava_vienkarsa.tif")
masketa_ainava=terra::mask(rastrs_ainava,
                           template_t,
                           filename="./RasterGrids_10m/2024/Ainava_vienk_mask
    ↪ .tif",
                           overwrite=TRUE)
plot(masketa_ainava)
```

```
# cleaning
rm(rastrs_ainava)
rm(masketa_ainava)
```

5.4 Landscape diversity

This subsection summarizes the input products related to the landscape described in the previous section – raster layers prepared with a resolution of 10 m, which characterize the classes found in the landscape (environment) and the subsequent preprocessing for the preparation of the EGV.

Shannon diversity index calculations are so resource-intensive that it is not rationally possible to perform them at every landscape scale around each analysis cell. They cannot be directly aggregated to speed up the calculation. Therefore, a decision has been made on the raster cell size, which:

- is formed as a multiplication of the analysis cell by an integer;
- is large enough to allow for environmental variability. Therefore, the analysis cell itself (or multiplication by 1) is not suitable - there is very little variability in land cover and land use in an area of 1 ha. This means that this cell must be as large as possible, but an analysis cell that is too large would mean an artificial intensification of spatial autocorrelation and a loss of spatial relevance;
- any landscape scale must be formed from several diversity-index-level cells.

Since we will use spatially weighted zonal statistics in the preparation of EGVs and the smallest landscape scale is r=500 m around the center of the EGV-cell, it has been decided to calculate the landscape diversity index for individual cells with a side length of 500 m, i.e., 25 ha landscapes. This means that the smallest number of units used for the development of the EGVs is nine (for a landscape scale of r=500 m around the center of the EGV-cell).

Three principal environments are described using diversity indices: landscape in general, farmland and forests. To make them easier to reproduce and find, each one is described in a separate section below.

5.4.1 Landscape in general diversity

Combination of three layers is involved to describe landscape in general diversity:

- as the lowest in hierarchy is `Ainava_vienk_mask.tif`, prepared in section Landscape classification;

- farmland diversity as a top layer in hierarhy. Prepared based on relatively broadly classified agricultural codes (field - SDM_grupa_sakums) from ural Support Service's information on declared fields. Only cells at declared fields have values, others are empty to be inherited from other layers during overlay. Codes used range from 351 to 362;
- forest diversity as a second layer in hierarchy. This layer describes tree species groups dominant in every stand with stand-level inventory interacted with age group as used in forestry practice. Values used in this classification are available from database description.
  - tree species groups:
    - coniferous species codes: "1", "3", "13", "14", "15", "22", "23";
    - boreal deciduous species codes: "4", "6", "8", "9", "19", "20", "21", "32", "35", "50", "68";
    - temperate dciduous species codes: "10", "11", "12", "16", "17", "18", "24", "25", "26", "27", "28", "29", "61", "62", "63", "64", "65", "66", "67", "69";
    - classification: forest is considered coniferous, if timber volume of coniferous species in top tree layer is at least 75% of total timer volume, else it can be considered boreal deciduous if the respective proportion is at least 75%, else it can be considered temperate deciduous if the respective proportion is 50%, else it is considered mixed.
  - tree age groups:
    - forests are considered young, if they are registered with age groups "1", "2" or "3";
    - forests are considered old, if they are registered with age groups "4", or "5";
  - created codes are formatted as factors and again as scalars with 660 added.

Once the landscape classification is done, diversity index is calculated at 25 ha landscapes with function `egvtools::landscape_function`. To guard value coverage, inverse distance weighted (power=2) gap filling is incorporated, however, there were no gaps to fill.

```
# Libs ----
if(!require(egvtools)) {install.packages("egvtools"); require(egvtools)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
```

```
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)

# general diversity ----

## Farmland broad ----

# classification
culturecodes=read_excel("./Geodata/2024/LAD/KulturuKodi_2024.xlsx")
culturecodes$kods=as.character(culturecodes$kods)
lad=sfarrow::st_read_parquet("./Geodata/2024/LAD/Lauki_2024.parquet")
lad2=lad %>%
  left_join(culturecodes, by=c("PRODUCT_CODE"="kods")) %>%
  mutate(numeric_code=as.numeric(as.factor(SDM_grupa_sakums))+350) %>%
  filter(!is.na(numeric_code))
table(lad2$numeric_code,useNA = "always")

# input layer
polygon2input(vector_data = lad2,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = "Diversity_FarmlandBroad_only.tif",
              value_field = "numeric_code",
              fun="first",
              prepare=FALSE,
              project_mode = "auto")
# cleaning
rm(culturecodes)
rm(lad)
rm(lad2)


## Forests broad ----

# data
mvr=sfarrow::st_read_parquet("./Geodata/2024/MVR/nogabali_2024janv.parquet")

# species groups
coniferous=c("1","3","13","14","15","22","23") # 7
boreal_deciduous=c("4","6","8","9","19","20","21","32","35","50","68") # 11
temperate_deciduous=c("10","11","12","16","17","18","24","25","26","27","28",
    ↪ "29",
            "61","62","63","64","65","66","67","69") # 20

# classification
mvr2=mvr %>%
  mutate(vol_coniferous=ifelse(s10 %in% coniferous,v10,0)+
           ifelse(s11 %in% coniferous,v11,0)+ifelse(s12 %in% coniferous,v12
    ↪ ,0)+
           ifelse(s13 %in% coniferous,v13,0)+ifelse(s14 %in% coniferous,v14
```

```
     ↪ ,0),
          vol_boreal=ifelse(s10 %in% boreal_deciduous,v10,0)+
            ifelse(s11 %in% boreal_deciduous,v11,0)+ifelse(s12 %in% boreal_
     ↪ deciduous,v12,0)+
            ifelse(s13 %in% boreal_deciduous,v13,0)+ifelse(s14 %in% boreal_
     ↪ deciduous,v14,0),
          vol_temperate=ifelse(s10 %in% temperate_deciduous,v10,0)+
            ifelse(s11 %in% temperate_deciduous,v11,0)+ifelse(s12 %in%
     ↪ temperate_deciduous,v12,0)+
            ifelse(s13 %in% temperate_deciduous,v13,0)+ifelse(s14 %in%
     ↪ temperate_deciduous,v14,0)) %>%
  mutate(vol_total=vol_coniferous+vol_boreal+vol_temperate) %>%
  mutate(forest_type=ifelse(vol_coniferous/vol_total>=0.75,"coniferous",
                   ifelse(vol_boreal/vol_total>=0.75,"boreal",
                          ifelse(vol_temperate/vol_total>0.5,"temperate",
                                 "mixed")))) %>%
  mutate(forest_age=ifelse(vgr=="1"|vgr=="2"|vgr=="3","young",
                          ifelse(vgr=="4"|vgr=="5","old",NA))) %>%
  filter(!is.na(forest_type)) %>%
  filter(!is.na(forest_age)) %>%
  mutate(divbroad_class=paste0(forest_type,"_",forest_age)) %>%
  mutate(divbroad_numeric=as.numeric(as.factor(divbroad_class))+660) %>%
  filter(!is.na(divbroad_numeric))

# input layer
polygon2input(vector_data = mvr2,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = "Diversity_ForestBroad_only.tif",
              value_field = "divbroad_numeric",
              fun="first",
              prepare=FALSE,
              project_mode = "auto",
              overwrite = TRUE)
# cleaning
rm(mvr)
rm(mvr2)

## General classification ----
simple_landscape=rast("./RasterGrids_10m/2024/Ainava_vienk_mask.tif")

## Covered classes for general diversity ----
farmland_broad=rast("./RasterGrids_10m/2024/Diversity_FarmlandBroad_only.tif"
     ↪ )
forests_broad=rast("./RasterGrids_10m/2024/Diversity_ForestBroad_only.tif")

diversity_classes=cover(farmland_broad,forests_broad)
diversity_classes2=cover(diversity_classes,simple_landscape,
                        filename="./RasterGrids_10m/2024/Diversity_
     ↪ GeneralLandscapeBroad.tif",
```

```
                         overwrite=TRUE)
rm(simple_landscape)
rm(farmland_broad)
rm(forests_broad)
rm(diversity_classes)
rm(diversity_classes2)

## Diversity index at 25ha -----
res_tbl <- landscape_function(
  landscape      = "./RasterGrids_10m/2024/Diversity_GeneralLandscapeBroad.
    ↪ tif",
  zones          = "./Templates/TemplateGrids/tikls500_sauzeme.parquet",
  id_field       = "rinda500",
  tile_field     = "tks50km",
  template       = "./Templates/TemplateRasters/LV500m_10km.tif",
  out_dir        = "./RasterGrids_500m/2024/",
  out_filename   = "Diversity_GeneralLandscape_500x.tif",
  out_layername  = "Diversity_GeneralLandscape_500x",
  what           = "lsm_l_shdi",
  rasterize_engine = "fasterize",
  n_workers      = 8,
  future_max_size = 3 * 1024^3,
  fill_gaps      = TRUE,
  plot_gaps      = TRUE,
  plot_result    = TRUE
)
print(res_tbl)
plot(rast("./RasterGrids_500m/2024/Diversity_GeneralLandscape_500x.tif"))
rm(res_tbl)
```

5.4.2 Forest diversity

An input grid with a cell size of 10 m covering the entire territory of Latvia. It contains the following values, in order of hierarchy:

- State Forest Service Forest State Register code, in which the code of the dominant tree species is multiplied by 1000 and the age group code is added. However, before rasterization, geometries in which no code has been created or one of the code components is 0 are excluded;
- forest diversity class values prepared in Landscape in general diversity;
- value 1 - other cells located in the territory of Latvia.

Once the landscape classification is done, diversity index is calculated at 25 ha landscapes with function `egvtools::landscape_function`. To guard value coverage, inverse distance weighted (power=2) gap filling is incorporated, however, there were no gaps to fill.

```
# Libs ----
if(!require(egvtools)) {install.packages("egvtools"); require(egvtools)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)

# forest diversity ----

## forest broad ----
forest_broad=rast("./RasterGrids_10m/2024/Diversity_ForestBroad_only.tif")


## forest codes ----

# žmei
mvr=st_read_parquet("./Geodata/2024/MVR/")

mvr=mvr %>%
  mutate(kods1=as.numeric(s10)*1000,
         kods2=as.numeric(vgr),
         kods=kods1+kods2) %>%
  filter(!is.na(kods)) %>%
  filter(kods1>0) %>%
  filter(kods2>0)

# input layer
polygon2input(vector_data = mvr,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = "Diversity_ForestCodes_only.tif",
              value_field = "kods",
              fun="first",
              prepare=FALSE,
              project_mode = "auto",
              overwrite = TRUE)

# cleaning
rm(mvr)

## Covered classes for forest diversity ----
```

```
forest_codes=rast("./RasterGrids_10m/2024/Diversity_ForestCodes_only.tif")
forest_covered=cover(forest_codes,forest_broad)
forest_covered2=cover(forest_covered,template_t,
                        filename="./RasterGrids_10m/2024/Diversity_
    ↪ ForestsDetailed.tif",
                        overwrite=TRUE)

# cleaning
rm(forest_codes)
rm(forest_covered)
rm(forest_covered2)
rm(forest_broad)



## Diversity index at 25ha -----
res_tbl <- landscape_function(
  landscape       = "./RasterGrids_10m/2024/Diversity_ForestsDetailed.tif",
  zones           = "./Templates/TemplateGrids/tikls500_sauzeme.parquet",
  id_field        = "rinda500",
  tile_field      = "tks50km",
  template        = "./Templates/TemplateRasters/LV500m_10km.tif",
  out_dir         = "./RasterGrids_500m/2024/",
  out_filename    = "Diversity_Forests_500x.tif",
  out_layername   = "Diversity_Forests_500x",
  what            = "lsm_l_shdi",
  rasterize_engine = "fasterize",
  n_workers       = 8,
  future_max_size = 3 * 1024^3,
  fill_gaps       = TRUE,
  plot_gaps       = TRUE,
  plot_result     = TRUE
)
print(res_tbl)

plot(rast("./RasterGrids_500m/2024/Diversity_Forests_500x.tif"))
rm(res_tbl)
```

5.4.3 Farmland diversity

A grid with a cell size of 10 m covering the entire territory of Latvia. It contains the following values, in order of hierarchy:

- Rural Support Service crop codes with 1000 added;
- farmland diversity class values prepared in Landscape in general diversity;
- value 1 - other cells located in the territory of Latvia.

Once the landscape classification is done, diversity index is calculated at 25 ha landscapes with function `egvtools::landscape_function`. To guard value coverage, inverse distance weighted (power=2) gap filling is incorporated, however, there were no gaps to fill.

```
# Libs ----
if(!require(egvtools)) {install.packages("egvtools"); require(egvtools)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(arrow)) {install.packages("arrow"); require(arrow)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}

# templates ----
template_t=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template_r=raster(template_t)


# farmland diversity -----

## Farmland broad ----
farmland_broad=rast("./RasterGrids_10m/2024/Diversity_FarmlandBroad_only.tif"
    ↪ )

## Farmland codes ----
lad=sfarrow::st_read_parquet("./Geodata/2024/LAD/Lauki_2024.parquet")
lad$product_code=as.numeric(lad$PRODUCT_CODE)+1000

# input layer
polygon2input(vector_data = lad,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = "Diversity_FarmlandCodes_only.tif",
              value_field = "product_code",
              fun="first",
              prepare=FALSE,
              project_mode = "auto",
              overwrite = TRUE)

# cleaning
rm(lad)


## Covered classes for farmland diversity ----
farmland_codes=rast("./RasterGrids_10m/2024/Diversity_FarmlandCodes_only.tif"
    ↪ )
farmland_covered=cover(farmland_codes,farmland_broad)
farmland_covered2=cover(farmland_covered,template_t,
                        filename="./RasterGrids_10m/2024/Diversity_
```

```
    ↪ FarmlandDetailed.tif",
                        overwrite=TRUE)
plot(farmland_covered2)

# cleaning
rm(farmland_codes)
rm(farmland_covered)
rm(farmland_covered2)
rm(farmland_broad)


## Diversity index at 25ha -----
res_tbl <- landscape_function(
  landscape      = "./RasterGrids_10m/2024/Diversity_FarmlandDetailed.tif",
  zones          = "./Templates/TemplateGrids/tikls500_sauzeme.parquet",
  id_field       = "rinda500",
  tile_field     = "tks50km",
  template       = "./Templates/TemplateRasters/LV500m_10km.tif",
  out_dir        = "./RasterGrids_500m/2024/",
  out_filename   = "Diversity_Farmland_500x.tif",
  out_layername  = "Diversity_Farmland_500x",
  what           = "lsm_l_shdi",
  rasterize_engine = "fasterize",
  n_workers      = 8,
  future_max_size = 3 * 1024^3,
  fill_gaps      = TRUE,
  plot_gaps      = TRUE,
  plot_result    = TRUE
)
print(res_tbl)

plot(rast("./RasterGrids_500m/2024/Diversity_Farmland_500x.tif"))
rm(res_tbl)
```

Chapter 6

Ecogeographical variables

Creation procedures of every EGV.

6.1 Climate_CHELSAv2.1-bio1_cell

filename: `Climate_CHELSAv2.1-bio1_cell.tif`

layername: `egv_1`

English name: Mean annual daily mean air temperature (°C) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējā ikdienas gaisa temperatūra (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio1_cell.tif"
layername="egv_1"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio1_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
```

```
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.2 Climate_CHELSAv2.1-bio10_cell

filename: `Climate_CHELSAv2.1-bio10_cell.tif`

layername: `egv_2`

English name: Mean daily mean air temperatures (°C) of the warmest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada siltākā ceturkšņa vidējā gaisa temperatūra (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio10_cell.tif"
layername="egv_2"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio10_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.3 Climate_CHELSAv2.1-bio11_cell

filename: `Climate_CHELSAv2.1-bio11_cell.tif`

layername: `egv_3`

English name: Mean daily mean air temperatures (°C) of the coldest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada aukstākā ceturkšņa vidējā gaisa temperatūra (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio11_cell.tif"
layername="egv_3"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio11_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.4 Climate_CHELSAv2.1-bio12_cell

filename: `Climate_CHELSAv2.1-bio12_cell.tif`

layername: `egv_4`

English name: Annual precipitation amount (kg m□^{2} year□^{1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada nokrišņu daudzums (kg m□² gadā) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio12_cell.tif"
layername="egv_4"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio12_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.5 Climate_CHELSAv2.1-bio13_cell

filename: `Climate_CHELSAv2.1-bio13_cell.tif`

layername: `egv_5`

English name: Precipitation amount (kg m□² month□¹) of the wettest month (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Slapjākā mēneša nokrišņu daudzums (kg m□² mēnesī) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# job ----

localname="Climate_CHELSAv2.1-bio13_cell.tif"
layername="egv_5"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio13_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.6 Climate_CHELSAv2.1-bio14_cell

filename: `Climate_CHELSAv2.1-bio14_cell.tif`

layername: `egv_6`

English name: Precipitation amount (kg m^{-2} month^{-1}) of the driest month (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Sausākā mēneša nokrišņu daudzums (kg m^{-2} mēnesī) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio14_cell.tif"
layername="egv_6"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio14_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
```

```
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.7 Climate_CHELSAv2.1-bio15_cell

filename: `Climate_CHELSAv2.1-bio15_cell.tif`

layername: `egv_7`

English name: Precipitation seasonality (kg m□²) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Nokrišņu sezonalitāte (kg m□²) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio15_cell.tif"
layername="egv_7"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio15_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.8 Climate_CHELSAv2.1-bio16_cell

filename: `Climate_CHELSAv2.1-bio16_cell.tif`

layername: `egv_8`

English name: Mean monthly precipitation amount (kg m^{-2} month^{-1}) of the wettest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Slapjākā ceturkšņa vidējais nokrišņu daudzums mēnesī (kg m^{-2} mēnesī) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```r
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio16_cell.tif"
layername="egv_8"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio16_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.9 Climate_CHELSAv2.1-bio17_cell

filename: `Climate_CHELSAv2.1-bio17_cell.tif`

layername: `egv_9`

English name: Mean monthly precipitation amount (kg m^{-2} month^{-1}) of the driest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Sausākā ceturkšņa vidējais nokrišņu daudzums mēnesī (kg m$^{\square 2}$ mēnesī) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio17_cell.tif"
layername="egv_9"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio17_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.10 Climate_CHELSAv2.1-bio18_cell

filename: `Climate_CHELSAv2.1-bio18_cell.tif`

layername: `egv_10`

English name: Mean monthly precipitation amount (kg m$^{\square 2}$ month$^{\square 1}$) of the warmest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Siltākā ceturkšņa vidējais nokrišņu daudzuma mēnesī (kg m$^{\square 2}$ mēnesī) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# job ----

localname="Climate_CHELSAv2.1-bio18_cell.tif"
layername="egv_10"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio18_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.11 Climate_CHELSAv2.1-bio19_cell

filename: `Climate_CHELSAv2.1-bio19_cell.tif`

layername: `egv_11`

English name: Mean monthly precipitation amount (kg m^{-2} month^{-1}) of the coldest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Aukstākā ceturkšņa vidējais nokrišņu daudzums mēnesī (kg m^{-2} mēnesī) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio19_cell.tif"
layername="egv_11"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio19_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
```

```
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.12 Climate_CHELSAv2.1-bio2_cell

filename: `Climate_CHELSAv2.1-bio2_cell.tif`

layername: `egv_12`

English name: Mean diurnal air temperature range (°C) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Diennakts temperatūru amplitūda (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio2_cell.tif"
layername="egv_12"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio2_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
```

```
print(df)
```

6.13 Climate_CHELSAv2.1-bio3_cell

filename: `Climate_CHELSAv2.1-bio3_cell.tif`

layername: `egv_13`

English name: Isothermality (ratio of diurnal variation to annual variation in temperatures) (°C) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Izotermalitāte (attiecība starp diennakts un gada temperatūras svārstībām) (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio3_cell.tif"
layername="egv_13"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio3_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.14 Climate_CHELSAv2.1-bio4_cell

filename: `Climate_CHELSAv2.1-bio4_cell.tif`

layername: `egv_14`

English name: Temperature seasonality (standard deviation of the monthly mean temperatures) (°C/100) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Temperatūru sezonalitāte (mēneša vidējo temperatūru standartnovirze) (°C/100) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio4_cell.tif"
layername="egv_14"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio4_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.15 Climate_CHELSAv2.1-bio5_cell

filename: `Climate_CHELSAv2.1-bio5_cell.tif`

layername: `egv_15`

English name: Mean daily maximum air temperature (°C) of the warmest month (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Siltākā mēneša vidējā ikdienas augstākā gaisa temperatūra (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio5_cell.tif"
layername="egv_15"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio5_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.16 Climate_CHELSAv2.1-bio6_cell

filename: `Climate_CHELSAv2.1-bio6_cell.tif`

layername: `egv_16`

English name: Mean daily minimum air temperature (°C) of the coldest month (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Aukstākā mēneša vidējā ikdienas zemākā gaisa temperatūra (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio6_cell.tif"
layername="egv_16"
```

```
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio6_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.17 Climate_CHELSAv2.1-bio7_cell

filename: `Climate_CHELSAv2.1-bio7_cell.tif`

layername: `egv_17`

English name: Annual range of air temperature (°C) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada temperatūru amplitūda (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio7_cell.tif"
layername="egv_17"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio7_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
```

```
  smooth          = TRUE,
  smooth_radius_km = 5,
  plot_result     = TRUE)
print(df)
```

6.18 Climate_CHELSAv2.1-bio8_cell

filename: `Climate_CHELSAv2.1-bio8_cell.tif`

layername: `egv_18`

English name: Mean daily mean air temperatures (°C) of the wettest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Slapjākā ceturkšņa vidējā ikdienas vidējā gaisa temperatūra (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio8_cell.tif"
layername="egv_18"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio8_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.19 Climate_CHELSAv2.1-bio9_cell

filename: `Climate_CHELSAv2.1-bio9_cell.tif`

layername: `egv_19`

English name: Mean daily mean air temperatures (°C) of the driest quarter (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Sausākā ceturkšņa vidējā ikdienas vidējā gaisa temperatūra (°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-bio9_cell.tif"
layername="egv_19"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-bio9_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.20 Climate_CHELSAv2.1-clt-max_cell

filename: `Climate_CHELSAv2.1-clt-max_cell.tif`

layername: `egv_20`

English name: Maximum monthly cloud area fraction (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Maksimālais mēneša vidējais mākoņu segums (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-clt-max_cell.tif"
layername="egv_20"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-clt-max_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.21 Climate_CHELSAv2.1-clt-mean_cell

filename: `Climate_CHELSAv2.1-clt-mean_cell.tif`

layername: `egv_21`

English name: Mean monthly cloud area fraction (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējais mākoņu segums (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# job ----

localname="Climate_CHELSAv2.1-clt-mean_cell.tif"
layername="egv_21"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-clt-mean_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.22 Climate_CHELSAv2.1-clt-min_cell

filename: `Climate_CHELSAv2.1-clt-min_cell.tif`

layername: `egv_22`

English name: Minimum monthly cloud area fraction (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Minimālais mēneša vidējais mākoņu segums (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-clt-min_cell.tif"
layername="egv_22"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-clt-min_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
```

```
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.23 Climate_CHELSAv2.1-clt-range_cell

filename: `Climate_CHELSAv2.1-clt-range_cell.tif`

layername: `egv_23`

English name: Annual range of monthly cloud area fraction (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada mākoņu seguma amplitūda (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-clt-range_cell.tif"
layername="egv_23"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-clt-range_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
```

```
print(df)
```

6.24 Climate_CHELSAv2.1-cmi-max_cell

filename: `Climate_CHELSAv2.1-cmi-max_cell.tif`

layername: `egv_24`

English name: Maximum monthly climate moisture index (kg m□^{2} month□^{1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Maksimālais mēneša vidējais klimata mitruma indekss (kg m□^{2} month□^{1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require(egvtools)}

# job ----

localname="Climate_CHELSAv2.1-cmi-max_cell.tif"
layername="egv_24"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-cmi-max_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.25 Climate_CHELSAv2.1-cmi-mean_cell

filename: `Climate_CHELSAv2.1-cmi-mean_cell.tif`

layername: `egv_25`

English name: Mean monthly climate moisture index (kg m□^{2} month□^{1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējais klimata mitruma indekss (kg m□^{2} month□^{1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-cmi-mean_cell.tif"
layername="egv_25"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-cmi-mean_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.26 Climate_CHELSAv2.1-cmi-min_cell

filename: `Climate_CHELSAv2.1-cmi-min_cell.tif`

layername: `egv_26`

English name: Minimum monthly climate moisture index (kg m□^{2} month□^{1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Minimālais mēneša vidējais klimata mitruma indekss (kg m□^{2} month□^{1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-cmi-min_cell.tif"
layername="egv_26"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-cmi-min_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.27 Climate_CHELSAv2.1-cmi-range_cell

filename: `Climate_CHELSAv2.1-cmi-range_cell.tif`

layername: `egv_27`

English name: Annual range of monthly climate moisture index (kg m□^{2} month□^{1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada klimata mitruma indeksa amplitūda (kg m□^{2} month□^{1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-cmi-range_cell.tif"
layername="egv_27"
```

```
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-cmi-range_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.28 Climate_CHELSAv2.1-fcf_cell

filename: `Climate_CHELSAv2.1-fcf_cell.tif`

layername: `egv_28`

English name: Frost change frequency (number of events in which tmin or tmax go above or below 0°C) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Sasalšanas gadījumu biežums (zemākā vai augstākā temperatūra šķērso 0°C) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-fcf_cell.tif"
layername="egv_28"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-fcf_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
```

```
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.29 Climate_CHELSAv2.1-fgd_cell

filename: `Climate_CHELSAv2.1-fgd_cell.tif`

layername: `egv_29`

English name: First day of the growing season (TREELIM) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas pirmā diena (TREELIM) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-fgd_cell.tif"
layername="egv_29"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-fgd_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.30 Climate_CHELSAv2.1-gdd0_cell

filename: `Climate_CHELSAv2.1-gdd0_cell.tif`

layername: `egv_30`

English name: Growing degree days heat sum above 0°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Aktīvo temperatūru summa no 0°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gdd0_cell.tif"
layername="egv_30"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gdd0_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.31 Climate_CHELSAv2.1-gdd10_cell

filename: `Climate_CHELSAv2.1-gdd10_cell.tif`

layername: `egv_31`

English name: Growing degree days heat sum above 10°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Aktīvo temperatūru summa no 10°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gdd10_cell.tif"
layername="egv_31"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gdd10_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.32 Climate_CHELSAv2.1-gdd5_cell

filename: `Climate_CHELSAv2.1-gdd5_cell.tif`

layername: `egv_32`

English name: Growing degree days heat sum above 5°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Aktīvo temperatūru summa no 5°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# job ----

localname="Climate_CHELSAv2.1-gdd5_cell.tif"
layername="egv_32"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gdd5_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.33 Climate_CHELSAv2.1-gddlgd0_cell

filename: `Climate_CHELSAv2.1-gddlgd0_cell.tif`

layername: `egv_33`

English name: Last growing degree day above 0°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas pēdējā diena no 0°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gddlgd0_cell.tif"
layername="egv_33"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gddlgd0_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
```

```
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.34 Climate_CHELSAv2.1-gddlgd10_cell

filename: `Climate_CHELSAv2.1-gddlgd10_cell.tif`

layername: `egv_34`

English name: Last growing degree day above 10°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas pēdējā diena no 10°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gddlgd10_cell.tif"
layername="egv_34"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gddlgd10_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
```

```
print(df)
```

6.35 Climate_CHELSAv2.1-gddlgd5_cell

filename: `Climate_CHELSAv2.1-gddlgd5_cell.tif`

layername: `egv_35`

English name: Last growing degree day above 5°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas pēdējā diena no 5°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gddlgd5_cell.tif"
layername="egv_35"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gddlgd5_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.36 Climate_CHELSAv2.1-gdgfgd0_cell

filename: `Climate_CHELSAv2.1-gdgfgd0_cell.tif`

layername: `egv_36`

English name: First growing degree day above 0°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas pirmā diena no 0°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gdgfgd0_cell.tif"
layername="egv_36"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gdgfgd0_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.37 Climate_CHELSAv2.1-gdgfgd10_cell

filename: `Climate_CHELSAv2.1-gdgfgd10_cell.tif`

layername: `egv_37`

English name: First growing degree day above 10°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas pirmā diena no 10°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gdgfgd10_cell.tif"
layername="egv_37"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gdgfgd10_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.38 Climate_CHELSAv2.1-gdgfgd5_cell

filename: `Climate_CHELSAv2.1-gdgfgd5_cell.tif`

layername: `egv_38`

English name: First growing degree day above 5°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas pirmā diena no 5°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gdgfgd5_cell.tif"
layername="egv_38"
```

```
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gdgfgd5_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.39 Climate_CHELSAv2.1-gsl_cell

filename: `Climate_CHELSAv2.1-gsl_cell.tif`

layername: `egv_39`

English name: Length of the growing season (TREELIM) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonas garums (TREELIM) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gsl_cell.tif"
layername="egv_39"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gsl_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
```

```
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.40 Climate_CHELSAv2.1-gsp_cell

filename: `Climate_CHELSAv2.1-gsp_cell.tif`

layername: `egv_40`

English name: Accumulated precipitation amount (kg m□^{2} year□^{1}) on growing season days (TREELIM) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Veģetācijas sezonā (TREELIM) uzkrātais nokrišņu daudzums (kg m□^{2} year□^{1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require(egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gsp_cell.tif"
layername="egv_40"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gsp_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.41 Climate_CHELSAv2.1-gst_cell

filename: `Climate_CHELSAv2.1-gst_cell.tif`

layername: `egv_41`

English name: Mean temperature of the growing season (TREELIM) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējā ikdienas gaisa temperatūra (°C) veģetācijas sezonā (TREELIM) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-gst_cell.tif"
layername="egv_41"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-gst_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.42 Climate_CHELSAv2.1-hurs-max_cell

filename: `Climate_CHELSAv2.1-hurs-max_cell.tif`

layername: `egv_42`

English name: Maximum monthly near-surface relative humidity (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Maksimālais mēneša vidējais gaisa mitrums (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-hurs-max_cell.tif"
layername="egv_42"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-hurs-max_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.43 Climate_CHELSAv2.1-hurs-mean_cell

filename: `Climate_CHELSAv2.1-hurs-mean_cell.tif`

layername: `egv_43`

English name: Mean monthly near-surface relative humidity (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējais ikmēneša gaisa mitrums (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# job ----

localname="Climate_CHELSAv2.1-hurs-mean_cell.tif"
layername="egv_43"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-hurs-mean_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.44 Climate_CHELSAv2.1-hurs-min_cell

filename: `Climate_CHELSAv2.1-hurs-min_cell.tif`

layername: `egv_44`

English name: Minimum monthly near-surface relative humidity (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Minimālais mēneša vidējais gaisa mitrums (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-hurs-min_cell.tif"
layername="egv_44"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-hurs-min_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
```

```
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.45 Climate_CHELSAv2.1-hurs-range_cell

filename: `Climate_CHELSAv2.1-hurs-range_cell.tif`

layername: `egv_45`

English name: Annual range of monthly near-surface relative humidity (%) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada gaisa mitruma amplitūda (%) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-hurs-range_cell.tif"
layername="egv_45"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-hurs-range_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
```

```
print(df)
```

6.46 Climate_CHELSAv2.1-lgd_cell

filename: `Climate_CHELSAv2.1-lgd_cell.tif`

layername: `egv_46`

English name: Last day of the growing season (TREELIM) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Pēdējā veģetācijas sezonas diena (TREELIM) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-lgd_cell.tif"
layername="egv_46"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-lgd_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.47 Climate_CHELSAv2.1-ngd0_cell

filename: `Climate_CHELSAv2.1-ngd0_cell.tif`

layername: `egv_47`

English name: Number of days at which 2m air temperature > 0°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Dienu skaits, kurā gaisa temperatūra 2 m augstumā pārsniedz 0°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-ngd0_cell.tif"
layername="egv_47"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-ngd0_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.48 Climate_CHELSAv2.1-ngd10_cell

filename: `Climate_CHELSAv2.1-ngd10_cell.tif`

layername: `egv_48`

English name: Number of days at which 2m air temperature > 10°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Dienu skaits, kurā gaisa temperatūra 2 m augstumā pārsniedz 10°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-ngd10_cell.tif"
layername="egv_48"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-ngd10_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.49 Climate_CHELSAv2.1-ngd5_cell

filename: `Climate_CHELSAv2.1-ngd5_cell.tif`

layername: `egv_49`

English name: Number of days at which 2m air temperature > 5°C (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Dienu skaits, kurā gaisa temperatūra 2 m augstumā pārsniedz 5°C (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-ngd5_cell.tif"
layername="egv_49"
```

```
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-ngd5_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.50 Climate_CHELSAv2.1-npp_cell

filename: `Climate_CHELSAv2.1-npp_cell.tif`

layername: `egv_50`

English name: Net primary productivity (g C m□² year□¹) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Neto primārā produkcija (g C m□² year□¹) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-npp_cell.tif"
layername="egv_50"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-npp_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
```

```
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.51 Climate_CHELSAv2.1-pet-penman-max_cell

filename: `Climate_CHELSAv2.1-pet-penman-max_cell.tif`

layername: `egv_51`

English name: Maximum monthly potential evapotranspiration (kg m^{-2} month^{-1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Maksimālā mēneša potenciālā evapotranspirācija (kg m^{-2} month^{-1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require(egvtools)}

# job ----

localname="Climate_CHELSAv2.1-pet-penman-max_cell.tif"
layername="egv_51"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-pet-penman-max_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.52 Climate_CHELSAv2.1-pet-penman-mean_cell

filename: `Climate_CHELSAv2.1-pet-penman-mean_cell.tif`

layername: `egv_52`

English name: Mean monthly potential evapotranspiration (kg m^{-2} month^{-1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējā mēneša potenciālā evapotranspirācija (kg m^{-2} month^{-1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-pet-penman-mean_cell.tif"
layername="egv_52"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-pet-penman-mean_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.53 Climate_CHELSAv2.1-pet-penman-min_cell

filename: `Climate_CHELSAv2.1-pet-penman-min_cell.tif`

layername: `egv_53`

English name: Minimum monthly potential evapotranspiration (kg m^{-2} month^{-1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Minimālā mēneša vidējā potenciālā evapotranspirācija (kg m^{-2} month^{-1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-pet-penman-min_cell.tif"
layername="egv_53"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-pet-penman-min_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.54 Climate_CHELSAv2.1-pet-penman-range_cell

filename: `Climate_CHELSAv2.1-pet-penman-range_cell.tif`

layername: `egv_54`

English name: Annual range of monthly potential evapotranspiration (kg m^{-2} month^{-1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada potenciālā evapotranspirācijas amplitūda (kg m^{-2} month^{-1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    (egvtools)}
```

```
# job ----

localname="Climate_CHELSAv2.1-pet-penman-range_cell.tif"
layername="egv_54"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-pet-penman-range_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.55 Climate_CHELSAv2.1-rsds-max_cell

filename: `Climate_CHELSAv2.1-rsds-max_cell.tif`

layername: `egv_55`

English name: Maximum monthly surface downwelling shortwave flux in air (MJ m⁻² d⁻¹) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Maksimālā mēneša vidējā Zemes virsmu sasniedzošā saules radiācija (MJ m⁻² d⁻¹) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-rsds-max_cell.tif"
layername="egv_55"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-rsds-max_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
```

```
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path   = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.56 Climate_CHELSAv2.1-rsds-mean_cell

filename: `Climate_CHELSAv2.1-rsds-mean_cell.tif`

layername: `egv_56`

English name: Mean monthly surface downwelling shortwave flux in air (MJ m^{-2} d^{-1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējā Zemes virsmu sasniedzošā saules radiācija (MJ m^{-2} d^{-1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-rsds-mean_cell.tif"
layername="egv_56"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-rsds-mean_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path   = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
```

```
print(df)
```

6.57 Climate_CHELSAv2.1-rsds-min_cell

filename: `Climate_CHELSAv2.1-rsds-min_cell.tif`

layername: `egv_57`

English name: Minimum monthly surface shortwave flux in air (MJ m□² d□¹) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Minimālā mēneša vidējā Zemes virsmu sasniedzošā saules radiācija (MJ m□² d□¹) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-rsds-min_cell.tif"
layername="egv_57"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-rsds-min_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.58 Climate_CHELSAv2.1-rsds-range_cell

filename: `Climate_CHELSAv2.1-rsds-range_cell.tif`

layername: `egv_58`

English name: Annual range of monthly surface downwelling shortwave flux in air (MJ m$^{\square 2}$ d$^{\square 1}$) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada amplitūda Zemes virsmu sasniedzošajai saules radiācijai (MJ m$^{\square 2}$ d$^{\square 1}$) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-rsds-range_cell.tif"
layername="egv_58"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-rsds-range_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.59 Climate_CHELSAv2.1-scd_cell

filename: `Climate_CHELSAv2.1-scd_cell.tif`

layername: `egv_59`

English name: Number of days with snow cover (TREELIM) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Dienu ar sniega segu skaits (TREELIM) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-scd_cell.tif"
layername="egv_59"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-scd_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.60 Climate_CHELSAv2.1-sfcWind-max_cell

filename: `Climate_CHELSAv2.1-sfcWind-max_cell.tif`

layername: `egv_60`

English name: Maximum monthly near-surface wind speed (m s□¹) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Maksimālais mēneša vidējais piezemes slāņa vēja ātrums (m s□¹) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-sfcWind-max_cell.tif"
layername="egv_60"
```

```
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-sfcWind-max_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.61 Climate_CHELSAv2.1-sfcWind-mean_cell

filename: `Climate_CHELSAv2.1-sfcWind-mean_cell.tif`

layername: `egv_61`

English name: Mean monthly near-surface wind speed (m s$^{\square 1}$) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējais piezemes slāņa vēja ātrums (m s$^{\square 1}$) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-sfcWind-mean_cell.tif"
layername="egv_61"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-sfcWind-mean_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
```

```
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.62 Climate_CHELSAv2.1-sfcWind-min_cell

filename: `Climate_CHELSAv2.1-sfcWind-min_cell.tif`

layername: `egv_62`

English name: Minimum monthly near-surface wind speed (m s□¹) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Minimālais mēneša vidējais piezemes slāņa vēja ātrums (m s□¹) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-sfcWind-min_cell.tif"
layername="egv_62"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-sfcWind-min_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path      = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path       = "./RasterGrids_100m/2024/RAW/",
  file_name      = localname,
  layer_name     = layername,
  fill_gaps      = TRUE,
  smooth         = TRUE,
  smooth_radius_km = 5,
  plot_result    = TRUE)
print(df)
```

6.63 Climate_CHELSAv2.1-sfcWind-range_cell

filename: `Climate_CHELSAv2.1-sfcWind-range_cell.tif`

layername: `egv_63`

English name: Annual range of monthly near-surface wind speed (m s^{-1}) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada amplitūda vidējam piezemes slāņa vēja ātrumam (m s^{-1}) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-sfcWind-range_cell.tif"
layername="egv_63"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-sfcWind-range_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.64 Climate_CHELSAv2.1-swb_cell

filename: `Climate_CHELSAv2.1-swb_cell.tif`

layername: `egv_64`

English name: Site water balance (kg m^{-2} $year^{-1}$) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Ūdens bilance (kg m^{-2} $year^{-1}$) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-swb_cell.tif"
layername="egv_64"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-swb_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.65 Climate_CHELSAv2.1-swe_cell

filename: `Climate_CHELSAv2.1-swe_cell.tif`

layername: `egv_65`

English name: Snow water equivalent (kg m$^{□2}$ year$^{□1}$) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Ūdens ekvivalents sniegā (kg m$^{□2}$ year$^{□1}$) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----
```

```
localname="Climate_CHELSAv2.1-swe_cell.tif"
layername="egv_65"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-swe_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.66 Climate_CHELSAv2.1-vpd-max_cell

filename: `Climate_CHELSAv2.1-vpd-max_cell.tif`

layername: `egv_66`

English name: Maximum monthly vapor pressure deficit (Pa) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Maksimālais mēneša vidējais iztvaikošanas spiediena deficīts (Pa) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-vpd-max_cell.tif"
layername="egv_66"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-vpd-max_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
```

```
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.67 Climate_CHELSAv2.1-vpd-mean_cell

filename: `Climate_CHELSAv2.1-vpd-mean_cell.tif`

layername: `egv_67`

English name: Mean monthly vapor pressure deficit (Pa) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Vidējais iztvaikošanas spiediena deficīts (Pa) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-vpd-mean_cell.tif"
layername="egv_67"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-vpd-mean_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.68 Climate_CHELSAv2.1-vpd-min_cell

filename: `Climate_CHELSAv2.1-vpd-min_cell.tif`

layername: `egv_68`

English name: Minimum monthly vapor pressure deficit (Pa) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Minimālais mēneša vidējais iztvaikošanas spiediena deficīts (Pa) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-vpd-min_cell.tif"
layername="egv_68"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-vpd-min_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.69 Climate_CHELSAv2.1-vpd-range_cell

filename: `Climate_CHELSAv2.1-vpd-range_cell.tif`

layername: `egv_69`

English name: Annual range of monthly vapor pressure deficit (Pa) (CHELSA v2.1) within the analysis cell (1 ha)

Latvian name: Gada iztvaikošanas spiediena deficīta amplitūda (Pa) (CHELSA v2.1) analīzes šūnā (1 ha)

Procedure: Directly follows CHELSA v2.1. EGV is prepared with the workflow `egvtools::downscale2egv()` with inverse distance weighted (power = 2) gap filling and soft smoothing (power = 0.5) over 5 km radius of every cell.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# job ----

localname="Climate_CHELSAv2.1-vpd-range_cell.tif"
layername="egv_69"
reading="./Geodata/2024/CHELSA/Climate_CHELSAv2.1-vpd-range_cell.tif"

df <- downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = reading,
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = localname,
  layer_name    = layername,
  fill_gaps     = TRUE,
  smooth        = TRUE,
  smooth_radius_km = 5,
  plot_result   = TRUE)
print(df)
```

6.70 HydroClim_01-max_cell

filename: `HydroClim_01-max_cell.tif`

layername: `egv_70`

English name: Maximum per subcatchment upstream mean annual air temperature (°C) (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā vidējā gaisa temperatūra augštecē (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_01-max_cell.tif"
layername="egv_70"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
```

```
                  missing_job = "FillOutput",
                  input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                  outlocation = "./RasterGrids_100m/2024/RAW/",
                  outfilename = localname,
                  layername = layername,
                  idw_weight = 2,
                  plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.71 HydroClim_02-max_cell

filename: `HydroClim_02-max_cell.tif`

layername: `egv_71`

English name: Maximum per subcatchment upstream mean diurnal air temperature range (°C) (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā diennakts gaisa temperatūras amplitūda augštecē (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
```

```
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_02-max_cell.tif"
layername="egv_71"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.72 HydroClim_03-max_cell

filename: `HydroClim_03-max_cell.tif`

layername: egv_72

English name: Maximum per subcatchment upstream isothermality (ratio of diurnal variation to annual variation in temperatures) (°C) (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā izotermalitāte augštecē (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_03-max_cell.tif"
layername="egv_72"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
```

```
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.73 HydroClim_04-max_cell

filename: `HydroClim_04-max_cell.tif`

layername: `egv_73`

English name: Maximum per subcatchment upstream temperature seasonality (standard deviation of the monthly mean temperatures) (°C/100) (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā temperatūras sezonalitāte augštecē (°C/100) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
```

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_04-max_cell.tif"
layername="egv_73"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
```

```
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.74 HydroClim_05-max_cell

filename: `HydroClim_05-max_cell.tif`

layername: `egv_74`

English name: Maximum per subcatchment upstream mean daily maximum air temperature (°C) of the warmest month (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā augšteces dienas vidējā gaisa temperatūra siltākajā mēnesī (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract` ↪ `()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)
```

```
# job ----

localname="HydroClim_05-max_cell.tif"
layername="egv_74"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.75 HydroClim_06-min_cell

filename: `HydroClim_06-min_cell.tif`

layername: `egv_75`

English name: Minimum per subcatchment upstream mean daily minimum air temperature (°C) of the coldest month (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina minimālā augšteces dienas vidējā gaisa temperatūra vēsākajā mēnesī (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - min - are calculated (`exactextractr::exact_extract()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_06-min_cell.tif"
layername="egv_75"
summary_function="min"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
```

```
                check_na = FALSE,
                plot_result=FALSE,
                plot_gaps = FALSE,
                overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.76 HydroClim_07-max_cell

filename: `HydroClim_07-max_cell.tif`

layername: `egv_76`

English name: Maximum per subcatchment upstream annual range of air temperature (°C) (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā augšteces gada gaisa temperatūru amplitūda (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
```

```
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_07-max_cell.tif"
layername="egv_76"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez
```

```
unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.77 HydroClim_08-max_cell

filename: `HydroClim_08-max_cell.tif`

layername: `egv_77`

English name: Maximum per subcatchment upstream mean daily mean air temperatures (°C) of the wettest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā augšteces dienas vidējā gaisa temperatūra mitrākajā ceturksnī (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_08-max_cell.tif"
layername="egv_77"
summary_function="max"
```

```
slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.78 HydroClim_09-min_cell

filename: `HydroClim_09-min_cell.tif`

layername: `egv_78`

English name: Minimum per subcatchment upstream mean daily mean air temperatures (°C) of the driest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā augšteces dienas vidējā gaisa temperatūra sausākajā ceturksnī (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - min - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_09-min_cell.tif"
layername="egv_78"
summary_function="min"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)
```

```
egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.79 HydroClim_10-max_cell

filename: `HydroClim_10-max_cell.tif`

layername: `egv_79`

English name: Maximum per subcatchment upstream mean daily mean air temperatures (°C) of the warmest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā augšteces dienas vidējā gaisa temperatūra siltākajā ceturksnī (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract` ↪ `()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
```

```
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_10-max_cell.tif"
layername="egv_79"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.80 HydroClim_11-min_cell

filename: `HydroClim_11-min_cell.tif`

layername: `egv_80`

English name: Minimum per subcatchment upstream mean daily mean air temperatures (°C) of the coldest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālā augšteces dienas vidējā gaisa temperatūra vēsākajā ceturksnī (°C) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - min - are calculated (`exactextractr::exact_extract()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_11-min_cell.tif"
layername="egv_80"
summary_function="min"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)
```

```
polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.81 HydroClim_12-max_cell

filename: `HydroClim_12-max_cell.tif`

layername: `egv_81`

English name: Maximum per subcatchment upstream annual precipitation amount (kg m^{-2} year^{-1}) (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzums gadā (kg m^{-2} year^{-1}) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_12-max_cell.tif"
layername="egv_81"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
```

```
↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.82 HydroClim_13-max_cell

filename: `HydroClim_13-max_cell.tif`

layername: `egv_82`

English name: Maximum per subcatchment upstream precipitation amount (kg m□² year□¹) of the wettest month (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzums mitrākajā mēnesī (kg m□² year□¹) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]
```

```
level12=st_make_valid(level12)

# job ----

localname="HydroClim_13-max_cell.tif"
layername="egv_82"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.83 HydroClim_14-max_cell

filename: `HydroClim_14-max_cell.tif`

layername: `egv_83`

English name: Maximum per subcatchment upstream precipitation amount (kg m^{-2} $year^{-1}$) of the driest month (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzums sausākajā mēnesī (kg m^{-2} $year^{-1}$) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_14-max_cell.tif"
layername="egv_83"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
```

```
               value_type = "continuous",
               prepare=FALSE,
               project_mode = "auto",
               check_na = FALSE,
               plot_result=FALSE,
               plot_gaps = FALSE,
               overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.84 HydroClim_15-max_cell

filename: `HydroClim_15-max_cell.tif`

layername: `egv_84`

English name: Maximum per subcatchment upstream precipitation seasonality (kg m□²) (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzuma sezonalitāte (kg m□²) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
```

```
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_15-max_cell.tif"
layername="egv_84"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
```

```
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.85 HydroClim_16-max_cell

filename: `HydroClim_16-max_cell.tif`

layername: `egv_85`

English name: Maximum per subcatchment upstream mean monthly precipitation amount (kg m□² year□¹) of the wettest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzums mitrākajā ceturksnī (kg m□² year□¹) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----
```

```
localname="HydroClim_16-max_cell.tif"
layername="egv_85"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.86 HydroClim_17-max_cell

filename: `HydroClim_17-max_cell.tif`

layername: `egv_86`

English name: Maximum per subcatchment upstream mean monthly precipitation amount (kg m□² year□¹) of the driest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzums sausākajā ceturksnī (kg m□² year□¹) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_17-max_cell.tif"
layername="egv_86"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
```

```
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.87 HydroClim_18-max_cell

filename: `HydroClim_18-max_cell.tif`

layername: `egv_87`

English name: Maximum per subcatchment upstream mean monthly precipitation amount (kg m^{-2} year^{-1}) of the warmest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzums siltākajā ceturksnī (kg m^{-2} year^{-1}) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}
```

```
# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_18-max_cell.tif"
layername="egv_87"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)

polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.88 HydroClim_19-max_cell

filename: `HydroClim_19-max_cell.tif`

layername: `egv_88`

English name: Maximum per subcatchment upstream mean monthly precipitation amount (kg m□² year□¹) of the coldest quarter (HydroClim) within the analysis cell (1 ha)

Latvian name: Sateces apakšbaseina maksimālais augšteces nokrišņu daudzums vēsākajā ceturksnī (kg m□² year□¹) (HydroClim) analīzes šūnā (1 ha)

Procedure: Information - both basins and raster layers - from HydroClim data is used. First, basin CRS is transformed to epsg:3059. Then zonal statistics (per basin) with layer specific summary function - max - are calculated (`exactextractr::exact_extract ↪ ()`) and then rasterized with `egvtools::polygon2input()`. Once rasterized to input data, EGV is created with `egvtools::input2egv()`. To prevent from gaps at the edges, inderse distance weighted (power = 2) gap filling is implemented. To save disk space, intermediate input layer is unlinked.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(sfarrow)) {install.packages("sfarrow"); require(sfarrow)}
if(!require(exactextractr)) {install.packages("exactextractr"); require(
    ↪ exactextractr)}

# basins ----
level12=st_read("./Geodata/2024/HydroClim/hybas_lake_eu_lev01-12_v1c/hybas_
    ↪ lake_eu_lev12_v1c.shp")
grid_1km=sfarrow::st_read_parquet("./Templates/TemplateGrids/tikls1km_sauzeme
    ↪ .parquet")
grid_1km=st_transform(grid_1km,crs=3059)
level12=st_transform(level12,crs=3059)
level12=level12[grid_1km,,]

level12=st_make_valid(level12)

# job ----

localname="HydroClim_19-max_cell.tif"
layername="egv_88"
summary_function="max"

slanis=rast(paste0("./Geodata/2024/HydroClim/",localname))
level12$Hydro_values=exact_extract(slanis,level12,fun=summary_function)
```

```
polygon2input(vector_data = level12,
              template_path = "./Templates/TemplateRasters/LV10m_10km.tif",
              out_path = "./RasterGrids_10m/2024/",
              file_name = localname,
              value_field = "Hydro_values",
              fun="first",
              value_type = "continuous",
              prepare=FALSE,
              project_mode = "auto",
              check_na = FALSE,
              plot_result=FALSE,
              plot_gaps = FALSE,
              overwrite=TRUE)

egvrez=input2egv(input=paste0("./RasterGrids_10m/2024/",localname),
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 input_template = "./Templates/TemplateRasters/LV10m_10km.tif
    ↪ ",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = localname,
                 layername = layername,
                 idw_weight = 2,
                 plot_gaps = FALSE,plot_final = FALSE)
egvrez

unlink(paste0("./RasterGrids_10m/2024/",localname))
```

6.89 Distance_Builtup_cell

filename: `Distance_Builtup_cell.tif`

layername: `egv_89`

English name: Distance to Built-Up features, average within the analysis cell (1 ha)

Latvian name: Attālums līdz apbūvei, vidējais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.90 Distance_ForestInside_cell

filename: `Distance_ForestInside_cell.tif`

layername: `egv_90`

English name: Distance to Forest Edge Inside Forests, average within the analysis cell (1 ha)

Latvian name: Attālums līdz meža malai tā iekšienē, vidējais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.91 Distance_GrasslandPermanent_cell

filename: `Distance_GrasslandPermanent_cell.tif`

layername: `egv_91`

English name: Distance to Permanent Grasslands, average within the analysis cell (1 ha)

Latvian name: Attālums līdz ilggadīgiem zālājiem, vidējais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.92 Distance_Landfill_cell

filename: `Distance_Landfill_cell.tif`

layername: `egv_92`

English name: Distance to Landfills, average within the analysis cell (1 ha)

Latvian name: Attālums līdz atkritumu poligoniem, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows Waste and garbage disposal sites, landfills. 1. From the attachaed file read sheet "Poligoni";

2. Create an `sf` object (epsg:3059);

3. Rasterize and cover so that cells of interest are 1 and others are 0;

4. create an egv with `egvtools::distance2egv()`. Expect warning regarding nothing to do with aggregation. It is because `egvtools::distance2egv()` already operate at egv-template not the input-template resolution. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented.

```r
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}
if(!require(egvtools)) {install.packages("egvtools"); require(egvtools)}

# templates ----
template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")
nulls100=rast("./Templates/TemplateRasters/nulls_LV100m_10km.tif")

# Distance_Landfill_cell.tif egv_92 ----

# reading coordinates
landfills=read_excel("./Geodata/2024/GarbageWasteLandfills/Atkritumi.xlsx",
    ↪ sheet="Poligoni")
#sf object
landfills_sf=st_as_sf(landfills,coords=c("X","Y"),crs=3059)
# rasterize
landfills_rast=rasterize(landfills_sf,template100)
# raster to 1=Cell of interest, 0=background
landfills_bg=cover(landfills_rast,nulls100)

# create an egv
distegv=distance2egv(input = landfills_bg,
             template_egv = template100,
             values_as_one = 1,
             fill_gaps = TRUE, idw_weight = 2,
             outlocation = "RasterGrids_100m/2024/RAW/",
             outfilename = "Distance_Landfill_cell.tif",
             layername = "egv_92")
distegv
```

6.93 Distance_Sea_cell

filename: `Distance_Sea_cell.tif`

layername: `egv_93`

English name: Distance to Sea, average within the analysis cell (1 ha)

Latvian name: Attālums līdz jūrai, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows Latvian Exclusive Economic Zone polygon. 1. Read layer as `sf` object (it already is epsg:3059);

2. Rasterize and cover so that cells of interest are 1 and others are 0;

3. create an egv with `egvtools::distance2egv()`. {fasterize} does not write CRS with `WKT` from epsg-string. Therefore it is better to use `project_to_template_input=TRUE` and define input-template. However, the only difference is in how the CRS is stored, therefore this can ignored - distance will be calculated on the input CRS and only resulting layer will be projected to match egv-template (faster due to 10x aggregation of resolution). To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(egvtools)) {install.packages("egvtools"); require(egvtools)}
if(!require(raster)) {install.packages("raster"); require(raster)}
if(!require(fasterize)) {install.packages("fasterize"); require(fasterize)}


# templates ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
nulls10=rast("./Templates/TemplateRasters/nulls_LV10m_10km.tif")
rastrs10=raster::raster(template10)


# Distance_Sea_cell.tif egv_93 ----

# sea layer, sf
sea=st_read("./Geodata/2024/LV_EEZ/LV_EEZ.shp")

# quick rasterization
sea_r=fasterize(sea,rastrs10,field="LV_EEZ")
sea_rast=rast(sea_r)

# # raster to 1=Cell of interest, 0=background
sea_bg=cover(sea_rast,nulls10)

# create an egv
distegv=distance2egv(input = sea_bg,
                     template_egv = "./Templates/TemplateRasters/LV100m_10km.
    ↪ tif",
                     values_as_one = 1,
                     project_to_template_input=TRUE, # fasterize stores CRS
    ↪ differently
                     template_input=template10,
                     fill_gaps = TRUE, idw_weight = 2,
                     outlocation = "RasterGrids_100m/2024/RAW/",
                     outfilename = "Distance_Sea_cell.tif",
                     layername = "egv_93")
distegv
```

6.94 Distance_Trees_cell

filename: `Distance_Trees_cell.tif`

layername: `egv_94`

English name: Distance to Trees, average within the analysis cell (1 ha)

Latvian name: Attālums līdz kokiem, vidējais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.95 Distance_Waste_cell

filename: `Distance_Waste_cell.tif`

layername: `egv_95`

English name: Distance to Waste disposal sites, average within the analysis cell (1 ha)

Latvian name: Attālums līdz atkritumu šķirošanas un uzglabāšanas vietām, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows Waste and garbage disposal sites, landfills. 1. From the attachaed file read sheet "AtkritumuVietas" and clean names;

2. Create an `sf` object (epsg:3059);
3. Filter to non-deposit collection locations;
4. Rasterize and cover so that cells of interest are 1 and others are 0;
5. create an egv with `egvtools::distance2egv()`. Expect warning regarding nothing to do with aggregation. It is because `egvtools::distance2egv()` already operate at egv-template not the input-template resolution. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(sf)) {install.packages("sf"); require(sf)}
if(!require(tidyverse)) {install.packages("tidyverse"); require(tidyverse)}
if(!require(readxl)) {install.packages("readxl"); require(readxl)}
if(!require(egvtools)) {install.packages("egvtools"); require(egvtools)}

# templates ----
```

```
template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")
nulls100=rast("./Templates/TemplateRasters/nulls_LV100m_10km.tif")


# Distance_Waste_cell.tif egv_95 ----

# reading coordinates
waste=read_excel("./Geodata/2024/GarbageWasteLandfills/Atkritumi.xlsx",sheet=
    ↪ "AtkritumuVietas")
# cleaning names
waste2=janitor::clean_names(waste)
#sf object
waste_sf=st_as_sf(waste2,coords=c("y_koordinata_lks92_tm","x_koordinata_lks92
    ↪ _tm"),crs=3059)
# filtering to non-deposit
table(waste_sf$pienemsanas_vietas_tips)
waste_sf2=waste_sf %>%
  filter(!str_detect(pienemsanas_vietas_tips,"īDepozta"))
# rasterize
waste_rast=rasterize(waste_sf2,template100)
# raster to 1=Cell of interest, 0=background
wastw_bg=cover(waste_rast,nulls100)

# create an egv
distegv=distance2egv(input = wastw_bg,
                     template_egv = template100,
                     values_as_one = 1,
                     fill_gaps = TRUE, idw_weight = 2,
                     outlocation = "RasterGrids_100m/2024/RAW/",
                     outfilename = "Distance_Waste_cell.tif",
                     layername = "egv_95")
distegv
```

6.96 Distance_Water_cell

filename: `Distance_Water_cell.tif`

layername: `egv_96`

English name: Distance to Waterbodies, average within the analysis cell (1 ha)

Latvian name: Attālums līdz ūdenstilpēn, vidējais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.97 Distance_WaterInside_cell

filename: `Distance_WaterInside_cell.tif`

layername: `egv_97`

English name: Distance to Waterbody Edge Inside Waterbody, average within the analysis cell (1 ha)

Latvian name: Attālums līdz ūdenstilpes malai tās iekšienē, vidējais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.98 Diversity_Farmland_r500

filename: `Diversity_Farmland_r500.tif`

layername: `egv_98`

English name: Average farmland class α-diversity of 500 m grid cells within the 0.5 km landscape

Latvian name: Vidējā lauku ainavas klašu 500 m šūnu α-daudzveidība 0.5 km ainavā

Procedure:

```
# libs ----
```

6.99 Diversity_Farmland_r1250

filename: `Diversity_Farmland_r1250.tif`

layername: `egv_99`

English name: Average farmland class α-diversity of 500 m grid cells within the 1.25 km landscape

Latvian name: Vidējā lauku ainavas klašu 500 m šūnu α-daudzveidība 1.25 km ainavā

Procedure:

```
# libs ----
```

6.100 Diversity_Farmland_r3000

filename: `Diversity_Farmland_r3000.tif`

layername: `egv_100`

English name: Average farmland class α-diversity of 500 m grid cells within the 3 km landscape

Latvian name: Vidējā lauku ainavas klašu 500 m šūnu α-daudzveidība 3 km ainavā

Procedure:

```
# libs ----
```

6.101 Diversity_Farmland_r10000

filename: `Diversity_Farmland_r10000.tif`

layername: `egv_101`

English name: Average farmland class α-diversity of 500 m grid cells within the 10 km landscape

Latvian name: Vidējā lauku ainavas klašu 500 m šūnu α-daudzveidība 10 km ainavā

Procedure:

```
# libs ----
```

6.102 Diversity_Forest_r500

filename: `Diversity_Forest_r500.tif`

layername: `egv_102`

English name: Average forest class α-diversity of 500 m grid cells within the 0.5 km landscape

Latvian name: Vidējā mežu ainavas klašu 500 m šūnu α-daudzveidība 0.5 km ainavā

Procedure:

```
# libs ----
```

6.103 Diversity_Forest_r1250

filename: `Diversity_Forest_r1250.tif`

layername: `egv_103`

English name: Average forest class α-diversity of 500 m grid cells within the 1.25 km landscape

Latvian name: Vidējā mežu ainavas klašu 500 m šūnu α-daudzveidība 1.25 km ainavā

Procedure:

```
# libs ----
```

6.104 Diversity_Forest_r3000

filename: `Diversity_Forest_r3000.tif`

layername: `egv_104`

English name: Average forest class α-diversity of 500 m grid cells within the 3 km landscape

Latvian name: Vidējā mežu ainavas klašu 500 m šūnu α-daudzveidība 3 km ainavā

Procedure:

```
# libs ----
```

6.105 Diversity_Forest_r10000

filename: `Diversity_Forest_r10000.tif`

layername: `egv_105`

English name: Average forest class α-diversity of 500 m grid cells within the 10 km landscape

Latvian name: Vidējā mežu ainavas klašu 500 m šūnu α-daudzveidība 10 km ainavā

Procedure:

```
# libs ----
```

6.106 Diversity_Total_r500

filename: `Diversity_Total_r500.tif`

layername: `egv_106`

English name: Average combined landscape α-diversity of 500 m grid cells within the 0.5 km landscape

Latvian name: Vidējā visu ainavas klašu 500 m šūnu α-daudzveidība 0.5 km ainavā

Procedure:

```
# libs ----
```

6.107 Diversity_Total_r1250

filename: `Diversity_Total_r1250.tif`

layername: `egv_107`

English name: Average combined landscape α-diversity of 500 m grid cells within the 1.25 km landscape

Latvian name: Vidējā visu ainavas klašu 500 m šūnu α-daudzveidība 1.25 km ainavā

Procedure:

```
# libs ----
```

6.108 Diversity_Total_r3000

filename: `Diversity_Total_r3000.tif`

layername: `egv_108`

English name: Average combined landscape α-diversity of 500 m grid cells within the 3 km landscape

Latvian name: Vidējā visu ainavas klašu 500 m šūnu α-daudzveidība 3 km ainavā

Procedure:

```
# libs ----
```

6.109 Diversity_Total_r10000

filename: `Diversity_Total_r10000.tif`

layername: `egv_109`

English name: Average combined landscape α-diversity of 500 m grid cells within the 10 km landscape

Latvian name: Vidējā visu ainavas klašu 500 m šūnu α-daudzveidība 10 km ainavā

Procedure:

```
# libs ----
```

6.110 Edges_Bogs-Trees_cell

filename: `Edges_Bogs-Trees_cell.tif`

layername: `egv_110`

English name: Edge pixels of Bogs, Mires bordering with Trees within the analysis cell (1 ha)

Latvian name: Purvu malu ar kokiem garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.111 Edges_Bogs-Trees_r500

filename: `Edges_Bogs-Trees_r500.tif`

layername: `egv_111`

English name: Edge pixels of Bogs, Mires bordering with Trees within the 0.5 km landscape

Latvian name: Purvu malu ar kokiem garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.112 Edges_Bogs-Trees_r1250

filename: `Edges_Bogs-Trees_r1250.tif`

layername: `egv_112`

English name: Edge pixels of Bogs, Mires bordering with Trees within the 1.25 km landscape

Latvian name: Purvu malu ar kokiem garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.113 Edges_Bogs-Trees_r3000

filename: `Edges_Bogs-Trees_r3000.tif`

layername: `egv_113`

English name: Edge pixels of Bogs, Mires bordering with Trees within the 3 km landscape

Latvian name: Purvu malu ar kokiem garums 3 km ainavā

Procedure:

```
# libs ----
```

6.114 Edges_Bogs-Trees_r10000

filename: `Edges_Bogs-Trees_r10000.tif`

layername: `egv_114`

English name: Edge pixels of Bogs, Mires bordering with Trees within the 10 km landscape

Latvian name: Purvu malu ar kokiem garums 10 km ainavā

Procedure:

```
# libs ----
```

6.115 Edges_Bogs-Water_cell

filename: `Edges_Bogs-Water_cell.tif`

layername: `egv_115`

English name: Edge pixels of Bogs, Mires bordering with Water within the analysis cell (1 ha)

Latvian name: Purvu malu ar ūdeni garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.116 Edges_Bogs-Water_r500

filename: `Edges_Bogs-Water_r500.tif`

layername: `egv_116`

English name: Edge pixels of Bogs, Mires bordering with Water within the 0.5 km landscape

Latvian name: Purvu malu ar ūdeni garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.117 Edges_Bogs-Water_r1250

filename: `Edges_Bogs-Water_r1250.tif`

layername: `egv_117`

English name: Edge pixels of Bogs, Mires bordering with Water within the 1.25 km landscape

Latvian name: Purvu malu ar ūdeni garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.118 Edges_Bogs-Water_r3000

filename: `Edges_Bogs-Water_r3000.tif`

layername: `egv_118`

English name: Edge pixels of Bogs, Mires bordering with Water within the 3 km landscape

Latvian name: Purvu malu ar ūdeni garums 3 km ainavā

Procedure:

```
# libs ----
```

6.119 Edges_Bogs-Water_r10000

filename: `Edges_Bogs-Water_r10000.tif`

layername: `egv_119`

English name: Edge pixels of Bogs, Mires bordering with Water within the 10 km landscape

Latvian name: Purvu malu ar ūdeni garums 10 km ainavā

Procedure:

```
# libs ----
```

6.120 Edges_Farmland-Builtup_cell

filename: `Edges_Farmland-Builtup_cell.tif`

layername: `egv_120`

English name: Edge pixels of Farmland bordering with Built-Up areas within the analysis cell (1 ha)

Latvian name: Lauksaimniecības zemju malu ar apbūvi garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.121 Edges_Farmland-Builtup_r500

filename: `Edges_Farmland-Builtup_r500.tif`

layername: `egv_121`

English name: Edge pixels of Farmland bordering with Built-Up areas within the 0.5 km landscape

Latvian name: Lauksaimniecības zemju malu ar apbūvi garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.122 Edges_Farmland-Builtup_r1250

filename: `Edges_Farmland-Builtup_r1250.tif`

layername: `egv_122`

English name: Edge pixels of Farmland bordering with Built-Up areas within the 1.25 km landscape

Latvian name: Lauksaimniecības zemju malu ar apbūvi garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.123 Edges_Farmland-Builtup_r3000

filename: `Edges_Farmland-Builtup_r3000.tif`

layername: `egv_123`

English name: Edge pixels of Farmland bordering with Built-Up areas within the 3 km landscape

Latvian name: Lauksaimniecības zemju malu ar apbūvi garums 3 km ainavā

Procedure:

```
# libs ----
```

6.124 Edges_Farmland-Builtup_r10000

filename: `Edges_Farmland-Builtup_r10000.tif`

layername: `egv_124`

English name: Edge pixels of Farmland bordering with Built-Up areas within the 10 km landscape

Latvian name: Lauksaimniecības zemju malu ar apbūvi garums 10 km ainavā

Procedure:

```
# libs ----
```

6.125 Edges_Trees-Builtup_cell

filename: `Edges_Trees-Builtup_cell.tif`

layername: `egv_125`

English name: Edge pixels of Trees bordering with Built-Up areas within the analysis cell (1 ha)

Latvian name: Koku malu ar apbūvi garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.126 Edges_Trees-Builtup_r500

filename: `Edges_Trees-Builtup_r500.tif`

layername: `egv_126`

English name: Edge pixels of Trees bordering with Built-Up areas within the 0.5 km landscape

Latvian name: Koku malu ar apbūvi garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.127 Edges_Trees-Builtup_r1250

filename: `Edges_Trees-Builtup_r1250.tif`

layername: `egv_127`

English name: Edge pixels of Trees bordering with Built-Up areas within the 1.25 km landscape

Latvian name: Koku malu ar apbūvi garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.128 Edges_Trees-Builtup_r3000

filename: `Edges_Trees-Builtup_r3000.tif`

layername: `egv_128`

English name: Edge pixels of Trees bordering with Built-Up areas within the 3 km landscape

Latvian name: Koku malu ar apbūvi garums 3 km ainavā

Procedure:

```
# libs ----
```

6.129 Edges_Trees-Builtup_r10000

filename: `Edges_Trees-Builtup_r10000.tif`

layername: `egv_129`

English name: Edge pixels of Trees bordering with Built-Up areas within the 10 km landscape

Latvian name: Koku malu ar apbūvi garums 10 km ainavā

Procedure:

```
# libs ----
```

6.130 Edges_CropsFallow_cell

filename: `Edges_CropsFallow_cell.tif`

layername: `egv_130`

English name: Edge pixels of Cropland, Fallow land within the analysis cell (1 ha)

Latvian name: Aramzemju malu garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.131 Edges_CropsFallow_r500

filename: `Edges_CropsFallow_r500.tif`

layername: `egv_131`

English name: Edge pixels of Cropland, Fallow land within the 0.5 km landscape

Latvian name: Aramzemju malu garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.132 Edges_CropsFallow_r1250

filename: `Edges_CropsFallow_r1250.tif`

layername: `egv_132`

English name: Edge pixels of Cropland, Fallow land within the 1.25 km landscape

Latvian name: Aramzemju malu garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.133 Edges_CropsFallow_r3000

filename: `Edges_CropsFallow_r3000.tif`

layername: `egv_133`

English name: Edge pixels of Cropland, Fallow land within the 3 km landscape

Latvian name: Aramzemju malu garums 3 km ainavā

Procedure:

```
# libs ----
```

6.134 Edges_CropsFallow_r10000

filename: `Edges_CropsFallow_r10000.tif`

layername: `egv_134`

English name: Edge pixels of Cropland, Fallow land within the 10 km landscape

Latvian name: Aramzemju malu garums 10 km ainavā

Procedure:

```
# libs ----
```

6.135 Edges_FarmlandShrubs-Trees_cell

filename: `Edges_FarmlandShrubs-Trees_cell.tif`

layername: `egv_135`

English name: Edge pixels of Farmland, Clear-Cuts, Shrubs bordering with Trees within the analysis cell (1 ha)

Latvian name: Lauksaimniecības zemju, izcirtumu, krūmu malu ar kokiem garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.136 Edges_FarmlandShrubs-Trees_r500

filename: `Edges_FarmlandShrubs-Trees_r500.tif`

layername: `egv_136`

English name: Edge pixels of Farmland, Clear-Cuts, Shrubs bordering with Trees within the 0.5 km landscape

Latvian name: Lauksaimniecības zemju, izcirtumu, krūmu malu ar kokiem garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.137 Edges_FarmlandShrubs-Trees_r1250

filename: `Edges_FarmlandShrubs-Trees_r1250.tif`

layername: `egv_137`

English name: Edge pixels of Farmland, Clear-Cuts, Shrubs bordering with Trees within the 1.25 km landscape

Latvian name: Lauksaimniecības zemju, izcirtumu, krūmu malu ar kokiem garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.138 Edges_FarmlandShrubs-Trees_r3000

filename: `Edges_FarmlandShrubs-Trees_r3000.tif`

layername: `egv_138`

English name: Edge pixels of Farmland, Clear-Cuts, Shrubs bordering with Trees within the 3 km landscape

Latvian name: Lauksaimniecības zemju, izcirtumu, krūmu malu ar kokiem garums 3 km ainavā

Procedure:

```
# libs ----
```

6.139 Edges_FarmlandShrubs-Trees_r10000

filename: `Edges_FarmlandShrubs-Trees_r10000.tif`

layername: `egv_139`

English name: Edge pixels of Farmland, Clear-Cuts, Shrubs bordering with Trees within the 10 km landscape

Latvian name: Lauksaimniecības zemju, izcirtumu, krūmu malu ar kokiem garums 10 km ainavā

Procedure:

```
# libs ----
```

6.140 Edges_Grasslands_cell

filename: `Edges_Grasslands_cell.tif`

layername: `egv_140`

English name: Edge pixels of Grassland within the analysis cell (1 ha)

Latvian name: Zālāju malu garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.141 Edges_Grasslands_r500

filename: `Edges_Grasslands_r500.tif`

layername: `egv_141`

English name: Edge pixels of Grassland within the 0.5 km landscape

Latvian name: Zālāju malu garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.142 Edges_Grasslands_r1250

filename: `Edges_Grasslands_r1250.tif`

layername: `egv_142`

English name: Edge pixels of Grassland within the 1.25 km landscape

Latvian name: Zālāju malu garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.143 Edges_Grasslands_r3000

filename: `Edges_Grasslands_r3000.tif`

layername: `egv_143`

English name: Edge pixels of Grassland within the 3 km landscape

Latvian name: Zālāju malu garums 3 km ainavā

Procedure:

```
# libs ----
```

6.144 Edges_Grasslands_r10000

filename: `Edges_Grasslands_r10000.tif`

layername: `egv_144`

English name: Edge pixels of Grassland within the 10 km landscape

Latvian name: Zālāju malu garums 10 km ainavā

Procedure:

```
# libs ----
```

6.145 Edges_OldForests_cell

filename: `Edges_OldForests_cell.tif`

layername: `egv_145`

English name: Edge pixels of Forests Over Rotation Age within the analysis cell (1 ha)

Latvian name: Pieaugušo un pāraugušo mežaudžu malu garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.146 Edges_OldForests_r500

filename: `Edges_OldForests_r500.tif`

layername: `egv_146`

English name: Edge pixels of Forests Over Rotation Age within the 0.5 km landscape

Latvian name: Pieaugušo un pāraugušo mežaudžu malu garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.147 Edges_OldForests_r1250

filename: `Edges_OldForests_r1250.tif`

layername: `egv_147`

English name: Edge pixels of Forests Over Rotation Age within the 1.25 km landscape

Latvian name: Pieaugušo un pāraugušo mežaudžu malu garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.148 Edges_OldForests_r3000

filename: `Edges_OldForests_r3000.tif`

layername: `egv_148`

English name: Edge pixels of Forests Over Rotation Age within the 3 km landscape

Latvian name: Pieaugušo un pāraugušo mežaudžu malu garums 3 km ainavā

Procedure:

```
# libs ----
```

6.149 Edges_OldForests_r10000

filename: `Edges_OldForests_r10000.tif`

layername: `egv_149`

English name: Edge pixels of Forests Over Rotation Age within the 10 km landscape

Latvian name: Pieaugušo un pāraugušo mežaudžu malu garums 10 km ainavā

Procedure:

```
# libs ----
```

6.150 Edges_Roads_cell

filename: `Edges_Roads_cell.tif`

layername: `egv_150`

English name: Edge pixels of Roads within the analysis cell (1 ha)

Latvian name: Ceļu malu garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.151 Edges_Roads_r500

filename: `Edges_Roads_r500.tif`

layername: `egv_151`

English name: Edge pixels of Roads within the 0.5 km landscape

Latvian name: Ceļu malu garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.152 Edges_Roads_r1250

filename: `Edges_Roads_r1250.tif`

layername: `egv_152`

English name: Edge pixels of Roads within the 1.25 km landscape

Latvian name: Ceļu malu garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.153 Edges_Roads_r3000

filename: `Edges_Roads_r3000.tif`

layername: `egv_153`

English name: Edge pixels of Roads within the 3 km landscape

Latvian name: Ceļu malu garums 3 km ainavā

Procedure:

```
# libs ----
```

6.154 Edges_Roads_r10000

filename: `Edges_Roads_r10000.tif`

layername: `egv_154`

English name: Edge pixels of Roads within the 10 km landscape

Latvian name: Ceļu malu garums 10 km ainavā

Procedure:

```
# libs ----
```

6.155 Edges_Trees_cell

filename: `Edges_Trees_cell.tif`

layername: `egv_155`

English name: Edge pixels of Trees within the analysis cell (1 ha)

Latvian name: Koku malu garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.156 Edges_Trees_r500

filename: `Edges_Trees_r500.tif`

layername: `egv_156`

English name: Edge pixels of Trees within the 0.5 km landscape

Latvian name: Koku malu garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.157 Edges_Trees_r1250

filename: `Edges_Trees_r1250.tif`

layername: `egv_157`

English name: Edge pixels of Trees within the 1.25 km landscape

Latvian name: Koku malu garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.158 Edges_Trees_r3000

filename: `Edges_Trees_r3000.tif`

layername: `egv_158`

English name: Edge pixels of Trees within the 3 km landscape

Latvian name: Koku malu garums 3 km ainavā

Procedure:

```
# libs ----
```

6.159 Edges_Trees_r10000

filename: `Edges_Trees_r10000.tif`

layername: `egv_159`

English name: Edge pixels of Trees within the 10 km landscape

Latvian name: Koku malu garums 10 km ainavā

Procedure:

```
# libs ----
```

6.160 Edges_Water_cell

filename: `Edges_Water_cell.tif`

layername: `egv_160`

English name: Edge pixels of Water within the analysis cell (1 ha)

Latvian name: Ūdenstilpju malu garums nalīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.161 Edges_Water_r500

filename: `Edges_Water_r500.tif`

layername: `egv_161`

English name: Edge pixels of Water within the 0.5 km landscape

Latvian name: Ūdenstilpju malu garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.162 Edges_Water_r1250

filename: `Edges_Water_r1250.tif`

layername: `egv_162`

English name: Edge pixels of Water within the 1.25 km landscape

Latvian name: Ūdenstilpju malu garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.163 Edges_Water_r3000

filename: `Edges_Water_r3000.tif`

layername: `egv_163`

English name: Edge pixels of Water within the 3 km landscape

Latvian name: Ūdenstilpju malu garums 3 km ainavā

Procedure:

```
# libs ----
```

6.164 Edges_Water_r10000

filename: `Edges_Water_r10000.tif`

layername: `egv_164`

English name: Edge pixels of Water within the 10 km landscape

Latvian name: Ūdenstilpju malu garums 10 km ainavā

Procedure:

```
# libs ----
```

6.165 Edges_Water-Farmland_cell

filename: `Edges_Water-Farmland_cell.tif`

layername: `egv_165`

English name: Edge pixels of Water bordering with Farmland within the analysis cell (1 ha)

Latvian name: Ūdenstilpu malu ar lauksaimniecības zemēm garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.166 Edges_Water-Farmland_r500

filename: `Edges_Water-Farmland_r500.tif`

layername: `egv_166`

English name: Edge pixels of Water bordering with Farmland within the 0.5 km landscape

Latvian name: Ūdenstilpu malu ar lauksaimniecības zemēm garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.167 Edges_Water-Farmland_r1250

filename: `Edges_Water-Farmland_r1250.tif`

layername: `egv_167`

English name: Edge pixels of Water bordering with Farmland within the 1.25 km landscape

Latvian name: Ūdenstilpu malu ar lauksaimniecības zemēm garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.168 Edges_Water-Farmland_r3000

filename: `Edges_Water-Farmland_r3000.tif`

layername: `egv_168`

English name: Edge pixels of Water bordering with Farmland within the 3 km landscape

Latvian name: Ūdenstilpu malu ar lauksaimniecības zemēm garums 3 km ainavā

Procedure:

```
# libs ----
```

6.169 Edges_Water-Farmland_r10000

filename: `Edges_Water-Farmland_r10000.tif`

layername: `egv_169`

English name: Edge pixels of Water bordering with Farmland within the 10 km landscape

Latvian name: Ūdenstilpu malu ar lauksaimniecības zemēm garums 10 km ainavā

Procedure:

```
# libs ----
```

6.170 Edges_Water-Grassland_cell

filename: `Edges_Water-Grassland_cell.tif`

layername: `egv_170`

English name: Edge pixels of Water bordering with Grassland within the analysis cell (1 ha)

Latvian name: Ūdenstilpu malu ar zālājiem garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.171 Edges_Water-Grassland_r500

filename: `Edges_Water-Grassland_r500.tif`

layername: `egv_171`

English name: Edge pixels of Water bordering with Grassland within the 0.5 km landscape

Latvian name: Ūdenstilpu malu ar zālājiem garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.172 Edges_Water-Grassland_r1250

filename: `Edges_Water-Grassland_r1250.tif`

layername: `egv_172`

English name: Edge pixels of Water bordering with Grassland within the 1.25 km landscape

Latvian name: Ūdenstilpu malu ar zālājiem garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.173 Edges_Water-Grassland_r3000

filename: `Edges_Water-Grassland_r3000.tif`

layername: `egv_173`

English name: Edge pixels of Water bordering with Grassland within the 3 km landscape

Latvian name: Ūdenstilpu malu ar zālājiem garums 3 km ainavā

Procedure:

```
# libs ----
```

6.174 Edges_Water-Grassland_r10000

filename: `Edges_Water-Grassland_r10000.tif`

layername: `egv_174`

English name: Edge pixels of Water bordering with Grassland within the 10 km landscape

Latvian name: Ūdenstilpu malu ar zālājiem garums 10 km ainavā

Procedure:

```
# libs ----
```

6.175 Edges_ReedSedgeRushBeds-Water_cell

filename: `Edges_ReedSedgeRushBeds-Water_cell.tif`

layername: `egv_175`

English name: Edge pixels of Reed-, Sedge-, Rush- Beds bordering with Water within the analysis cell (1 ha)

Latvian name: Niedrāju, grīslāju, meldrāju malu ar ūdeni garums analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.176 Edges_ReedSedgeRushBeds-Water_r500

filename: `Edges_ReedSedgeRushBeds-Water_r500.tif`

layername: `egv_176`

English name: Edge pixels of Reed-, Sedge-, Rush- Beds bordering with Water within the 0.5 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju malu ar ūdeni garums 0,5 km ainavā

Procedure:

```
# libs ----
```

6.177 Edges_ReedSedgeRushBeds-Water_r1250

filename: `Edges_ReedSedgeRushBeds-Water_r1250.tif`

layername: `egv_177`

English name: Edge pixels of Reed-, Sedge-, Rush- Beds bordering with Water within the 1.25 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju malu ar ūdeni garums 1,25 km ainavā

Procedure:

```
# libs ----
```

6.178 Edges_ReedSedgeRushBeds-Water_r3000

filename: `Edges_ReedSedgeRushBeds-Water_r3000.tif`

layername: `egv_178`

English name: Edge pixels of Reed-, Sedge-, Rush- Beds bordering with Water within the 3 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju malu ar ūdeni garums 3 km ainavā

Procedure:

```
# libs ----
```

6.179 Edges_ReedSedgeRushBeds-Water_r10000

filename: `Edges_ReedSedgeRushBeds-Water_r10000.tif`

layername: `egv_179`

English name: Edge pixels of Reed-, Sedge-, Rush- Beds bordering with Water within the 10 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju malu ar ūdeni garums 10 km ainavā

Procedure:

```
# libs ----
```

6.180 FarmlandCrops_CropsAll_cell

filename: `FarmlandCrops_CropsAll_cell.tif`

layername: `egv_180`

English name: Fractional cover of Crops (all types) within the analysis cell (1 ha)

Latvian name: Aramzemju (dažādu lauksaimniecības kultūru) platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.181 FarmlandCrops_CropsAll_r500

filename: `FarmlandCrops_CropsAll_r500.tif`

layername: `egv_181`

English name: Fractional cover of Crops (all types) within the 0.5 km landscape

Latvian name: Aramzemju (dažādu lauksaimniecības kultūru) platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.182 FarmlandCrops_CropsAll_r1250

filename: `FarmlandCrops_CropsAll_r1250.tif`

layername: `egv_182`

English name: Fractional cover of Crops (all types) within the 1.25 km landscape

Latvian name: Aramzemju (dažādu lauksaimniecības kultūru) platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.183 FarmlandCrops_CropsAll_r3000

filename: `FarmlandCrops_CropsAll_r3000.tif`

layername: `egv_183`

English name: Fractional cover of Crops (all types) within the 3 km landscape

Latvian name: Aramzemju (dažādu lauksaimniecības kultūru) platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.184 FarmlandCrops_CropsAll_r10000

filename: `FarmlandCrops_CropsAll_r10000.tif`

layername: `egv_184`

English name: Fractional cover of Crops (all types) within the 10 km landscape

Latvian name: Aramzemju (dažādu lauksaimniecības kultūru) platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.185 FarmlandCrops_CropsHoed_cell

filename: `FarmlandCrops_CropsHoed_cell.tif`

layername: `egv_185`

English name: Fractional cover of Hoed Crops within the analysis cell (1 ha)

Latvian name: Vagu un rušināmkultūru platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.186 FarmlandCrops_CropsHoed_r500

filename: `FarmlandCrops_CropsHoed_r500.tif`

layername: `egv_186`

English name: Fractional cover of Hoed Crops within the 0.5 km landscape

Latvian name: Vagu un rušināmkultūru platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.187 FarmlandCrops_CropsHoed_r1250

filename: `FarmlandCrops_CropsHoed_r1250.tif`

layername: `egv_187`

English name: Fractional cover of Hoed Crops within the 1.25 km landscape

Latvian name: Vagu un rušināmkultūru platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.188 FarmlandCrops_CropsHoed_r3000

filename: `FarmlandCrops_CropsHoed_r3000.tif`

layername: `egv_188`

English name: Fractional cover of Hoed Crops within the 3 km landscape

Latvian name: Vagu un rušināmkultūru platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.189 FarmlandCrops_CropsHoed_r10000

filename: `FarmlandCrops_CropsHoed_r10000.tif`

layername: `egv_189`

English name: Fractional cover of Hoed Crops within the 10 km landscape

Latvian name: Vagu un rušināmkultūru platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.190 FarmlandCrops_CropsOther_cell

filename: `FarmlandCrops_CropsOther_cell.tif`

layername: `egv_190`

English name: Fractional cover of Other Crops within the analysis cell (1 ha)

Latvian name: Citu lauksaimniecības kultūraugu aramzemēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.191 FarmlandCrops_CropsOther_r500

filename: `FarmlandCrops_CropsOther_r500.tif`

layername: `egv_191`

English name: Fractional cover of Other Crops within the 0.5 km landscape

Latvian name: Citu lauksaimniecības kultūraugu aramzemēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.192 FarmlandCrops_CropsOther_r1250

filename: `FarmlandCrops_CropsOther_r1250.tif`

layername: `egv_192`

English name: Fractional cover of Other Crops within the 1.25 km landscape

Latvian name: Citu lauksaimniecības kultūraugu aramzemēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.193 FarmlandCrops_CropsOther_r3000

filename: `FarmlandCrops_CropsOther_r3000.tif`

layername: `egv_193`

English name: Fractional cover of Other Crops within the 3 km landscape

Latvian name: Citu lauksaimniecības kultūraugu aramzemēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.194 FarmlandCrops_CropsOther_r10000

filename: `FarmlandCrops_CropsOther_r10000.tif`

layername: `egv_194`

English name: Fractional cover of Other Crops within the 10 km landscape

Latvian name: Citu lauksaimniecības kultūraugu aramzemēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.195 FarmlandCrops_CropsSpring_cell

filename: `FarmlandCrops_CropsSpring_cell.tif`

layername: `egv_195`

English name: Fractional cover of Spring Sown Crops within the analysis cell (1 ha)

Latvian name: Vasarāju aramzemēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.196 FarmlandCrops_CropsSpring_r500

filename: `FarmlandCrops_CropsSpring_r500.tif`

layername: `egv_196`

English name: Fractional cover of Spring Sown Crops within the 0.5 km landscape

Latvian name: Vasarāju aramzemēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.197 FarmlandCrops_CropsSpring_r1250

filename: `FarmlandCrops_CropsSpring_r1250.tif`

layername: `egv_197`

English name: Fractional cover of Spring Sown Crops within the 1.25 km landscape

Latvian name: Vasarāju aramzemēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.198 FarmlandCrops_CropsSpring_r3000

filename: `FarmlandCrops_CropsSpring_r3000.tif`

layername: `egv_198`

English name: Fractional cover of Spring Sown Crops within the 3 km landscape

Latvian name: Vasarāju aramzemēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.199 FarmlandCrops_CropsSpring_r10000

filename: `FarmlandCrops_CropsSpring_r10000.tif`

layername: `egv_199`

English name: Fractional cover of Spring Sown Crops within the 10 km landscape

Latvian name: Vasarāju aramzemēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.200 FarmlandCrops_CropsWinter_cell

filename: `FarmlandCrops_CropsWinter_cell.tif`

layername: `egv_200`

English name: Fractional cover of Winter Crops within the analysis cell (1 ha)

Latvian name: Ziemāju aramzemēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.201 FarmlandCrops_CropsWinter_r500

filename: `FarmlandCrops_CropsWinter_r500.tif`

layername: `egv_201`

English name: Fractional cover of Winter Crops within the 0.5 km landscape

Latvian name: Ziemāju aramzemēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.202 FarmlandCrops_CropsWinter_r1250

filename: `FarmlandCrops_CropsWinter_r1250.tif`

layername: `egv_202`

English name: Fractional cover of Winter Crops within the 1.25 km landscape

Latvian name: Ziemāju aramzemēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.203 FarmlandCrops_CropsWinter_r3000

filename: `FarmlandCrops_CropsWinter_r3000.tif`

layername: `egv_203`

English name: Fractional cover of Winter Crops within the 3 km landscape

Latvian name: Ziemāju aramzemēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.204 FarmlandCrops_CropsWinter_r10000

filename: `FarmlandCrops_CropsWinter_r10000.tif`

layername: `egv_204`

English name: Fractional cover of Winter Crops within the 10 km landscape

Latvian name: Ziemāju aramzemēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.205 FarmlandCrops_RapeseedsSpring_cell

filename: `FarmlandCrops_RapeseedsSpring_cell.tif`

layername: `egv_205`

English name: Fractional cover of Spring Sown Rapeseed, Turnip, Corn within the analysis cell (1 ha)

Latvian name: Vasaras rapša, ripša, kukurūzas platība analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.206 FarmlandCrops_RapeseedsSpring_r500

filename: `FarmlandCrops_RapeseedsSpring_r500.tif`

layername: `egv_206`

English name: Fractional cover of Spring Sown Rapeseed, Turnip, Corn within the 0.5 km landscape

Latvian name: Vasaras rapša, ripša, kukurūzas platība 0,5 km ainavā

Procedure:

```
# libs ----
```

6.207 FarmlandCrops_RapeseedsSpring_r1250

filename: `FarmlandCrops_RapeseedsSpring_r1250.tif`

layername: `egv_207`

English name: Fractional cover of Spring Sown Rapeseed, Turnip, Corn within the 1.25 km landscape

Latvian name: Vasaras rapša, ripša, kukurūzas platība 1,25 km ainavā

Procedure:

```
# libs ----
```

6.208 FarmlandCrops_RapeseedsSpring_r3000

filename: `FarmlandCrops_RapeseedsSpring_r3000.tif`

layername: `egv_208`

English name: Fractional cover of Spring Sown Rapeseed, Turnip, Corn within the 3 km landscape

Latvian name: Vasaras rapša, ripša, kukurūzas platība 3 km ainavā

Procedure:

```
# libs ----
```

6.209 FarmlandCrops_RapeseedsSpring_r10000

filename: `FarmlandCrops_RapeseedsSpring_r10000.tif`

layername: `egv_209`

English name: Fractional cover of Spring Sown Rapeseed, Turnip, Corn within the 10 km landscape

Latvian name: Vasaras rapša, ripša, kukurūzas platība 10 km ainavā

Procedure:

```
# libs ----
```

6.210 FarmlandCrops_RapeseedsWinter_cell

filename: `FarmlandCrops_RapeseedsWinter_cell.tif`

layername: `egv_210`

English name: Fractional cover of Winter Rapeseed, Turnip within the analysis cell (1 ha)

Latvian name: Ziemas rapša, ripša platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.211 FarmlandCrops_RapeseedsWinter_r500

filename: `FarmlandCrops_RapeseedsWinter_r500.tif`

layername: `egv_211`

English name: Fractional cover of Winter Rapeseed, Turnip within the 0.5 km landscape

Latvian name: Ziemas rapša, ripša platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.212 FarmlandCrops_RapeseedsWinter_r1250

filename: `FarmlandCrops_RapeseedsWinter_r1250.tif`

layername: `egv_212`

English name: Fractional cover of Winter Rapeseed, Turnip within the 1.25 km landscape

Latvian name: Ziemas rapša, ripša platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.213 FarmlandCrops_RapeseedsWinter_r3000

filename: `FarmlandCrops_RapeseedsWinter_r3000.tif`

layername: `egv_213`

English name: Fractional cover of Winter Rapeseed, Turnip within the 3 km landscape

Latvian name: Ziemas rapša, ripša platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.214 FarmlandCrops_RapeseedsWinter_r10000

filename: `FarmlandCrops_RapeseedsWinter_r10000.tif`

layername: `egv_214`

English name: Fractional cover of Winter Rapeseed, Turnip within the 10 km landscape

Latvian name: Ziemas rapša, ripša platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.215 FarmlandGrassland_GrasslandsAbandoned_cell

filename: `FarmlandGrassland_GrasslandsAbandoned_cell.tif`

layername: `egv_215`

English name: Fractional cover of Abandoned Grassland within the analysis cell (1 ha)

Latvian name: Neapsaimniekotu zālāju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.216 FarmlandGrassland_GrasslandsAbandoned_r500

filename: `FarmlandGrassland_GrasslandsAbandoned_r500.tif`

layername: `egv_216`

English name: Fractional cover of Abandoned Grassland within the 0.5 km landscape

Latvian name: Neapsaimniekotu zālāju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.217 FarmlandGrassland_GrasslandsAbandoned_r1250

filename: `FarmlandGrassland_GrasslandsAbandoned_r1250.tif`

layername: `egv_217`

English name: Fractional cover of Abandoned Grassland within the 1.25 km landscape

Latvian name: Neapsaimniekotu zālāju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.218 FarmlandGrassland_GrasslandsAbandoned_r3000

filename: `FarmlandGrassland_GrasslandsAbandoned_r3000.tif`

layername: `egv_218`

English name: Fractional cover of Abandoned Grassland within the 3 km landscape

Latvian name: Neapsaimniekotu zālāju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.219 FarmlandGrassland_GrasslandsAbandoned_r10000

filename: `FarmlandGrassland_GrasslandsAbandoned_r10000.tif`

layername: `egv_219`

English name: Fractional cover of Abandoned Grassland within the 10 km landscape

Latvian name: Neapsaimniekotu zālāju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.220 FarmlandGrassland_GrasslandsAll_cell

filename: `FarmlandGrassland_GrasslandsAll_cell.tif`

layername: `egv_220`

English name: Fractional cover of any Grassland within the analysis cell (1 ha)

Latvian name: Zālāju (visu veidu) platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.221 FarmlandGrassland_GrasslandsAll_r500

filename: `FarmlandGrassland_GrasslandsAll_r500.tif`

layername: `egv_221`

English name: Fractional cover of any Grassland within the 0.5 km landscape

Latvian name: Zālāju (visu veidu) platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.222 FarmlandGrassland_GrasslandsAll_r1250

filename: `FarmlandGrassland_GrasslandsAll_r1250.tif`

layername: `egv_222`

English name: Fractional cover of any Grassland within the 1.25 km landscape

Latvian name: Zālāju (visu veidu) platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.223 FarmlandGrassland_GrasslandsAll_r3000

filename: `FarmlandGrassland_GrasslandsAll_r3000.tif`

layername: `egv_223`

English name: Fractional cover of any Grassland within the 3 km landscape

Latvian name: Zālāju (visu veidu) platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.224 FarmlandGrassland_GrasslandsAll_r10000

filename: `FarmlandGrassland_GrasslandsAll_r10000.tif`

layername: `egv_224`

English name: Fractional cover of any Grassland within the 10 km landscape

Latvian name: Zālāju (visu veidu) platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.225 FarmlandGrassland_GrasslandsPermanent_cell

filename: `FarmlandGrassland_GrasslandsPermanent_cell.tif`

layername: `egv_225`

English name: Fractional cover of Permanent Grassland within the analysis cell (1 ha)

Latvian name: Ilggadīgu zālāju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.226 FarmlandGrassland_GrasslandsPermanent_r500

filename: `FarmlandGrassland_GrasslandsPermanent_r500.tif`

layername: `egv_226`

English name: Fractional cover of Permanent Grassland within the 0.5 km landscape

Latvian name: Ilggadīgu zālāju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.227 FarmlandGrassland_GrasslandsPermanent_r1250

filename: `FarmlandGrassland_GrasslandsPermanent_r1250.tif`

layername: `egv_227`

English name: Fractional cover of Permanent Grassland within the 1.25 km landscape

Latvian name: Ilggadīgu zālāju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.228 FarmlandGrassland_GrasslandsPermanent_r3000

filename: `FarmlandGrassland_GrasslandsPermanent_r3000.tif`

layername: `egv_228`

English name: Fractional cover of Permanent Grassland within the 3 km landscape

Latvian name: Ilggadīgu zālāju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.229 FarmlandGrassland_GrasslandsPermanent_r10000

filename: `FarmlandGrassland_GrasslandsPermanent_r10000.tif`

layername: `egv_229`

English name: Fractional cover of Permanent Grassland within the 10 km landscape

Latvian name: Ilggadīgu zālāju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.230 FarmlandGrassland_GrasslandsTemporary_cell

filename: `FarmlandGrassland_GrasslandsTemporary_cell.tif`

layername: `egv_230`

English name: Fractional cover of Temporary Grassland within the analysis cell (1 ha)

Latvian name: Zālāju-aramzemē platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.231 FarmlandGrassland_GrasslandsTemporary_r500

filename: `FarmlandGrassland_GrasslandsTemporary_r500.tif`

layername: `egv_231`

English name: Fractional cover of Temporary Grassland within the 0.5 km landscape

Latvian name: Zālāju-aramzemē platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.232 FarmlandGrassland_GrasslandsTemporary_r1250

filename: `FarmlandGrassland_GrasslandsTemporary_r1250.tif`

layername: `egv_232`

English name: Fractional cover of Temporary Grassland within the 1.25 km landscape

Latvian name: Zālāju-aramzemē platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.233 FarmlandGrassland_GrasslandsTemporary_r3000

filename: `FarmlandGrassland_GrasslandsTemporary_r3000.tif`

layername: `egv_233`

English name: Fractional cover of Temporary Grassland within the 3 km landscape

Latvian name: Zālāju-aramzemē platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.234 FarmlandGrassland_GrasslandsTemporary_r10000

filename: `FarmlandGrassland_GrasslandsTemporary_r10000.tif`

layername: `egv_234`

English name: Fractional cover of Temporary Grassland within the 10 km landscape

Latvian name: Zālāju-aramzemē platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.235 FarmlandParcels_FieldsActive_cell

filename: `FarmlandParcels_FieldsActive_cell.tif`

layername: `egv_235`

English name: Fractional cover of Agricultural Land Parcels within the analysis cell (1 ha)

Latvian name: Lauku bloku platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.236 FarmlandParcels_FieldsActive_r500

filename: `FarmlandParcels_FieldsActive_r500.tif`

layername: `egv_236`

English name: Fractional cover of Agricultural Land Parcels within the 0.5 km landscape

Latvian name: Lauku bloku platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.237 FarmlandParcels_FieldsActive_r1250

filename: `FarmlandParcels_FieldsActive_r1250.tif`

layername: `egv_237`

English name: Fractional cover of Agricultural Land Parcels within the 1.25 km landscape

Latvian name: Lauku bloku platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.238 FarmlandParcels_FieldsActive_r3000

filename: `FarmlandParcels_FieldsActive_r3000.tif`

layername: `egv_238`

English name: Fractional cover of Agricultural Land Parcels within the 3 km landscape

Latvian name: Lauku bloku platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.239 FarmlandParcels_FieldsActive_r10000

filename: `FarmlandParcels_FieldsActive_r10000.tif`

layername: `egv_239`

English name: Fractional cover of Agricultural Land Parcels within the 10 km landscape

Latvian name: Lauku bloku platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.240 FarmlandPloughed_CropsFallow_cell

filename: `FarmlandPloughed_CropsFallow_cell.tif`

layername: `egv_240`

English name: Fractional cover of Crop-, Fallow- Land within the analysis cell (1 ha)

Latvian name: Aramzemju, papuvju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.241 FarmlandPloughed_CropsFallow_r500

filename: `FarmlandPloughed_CropsFallow_r500.tif`

layername: `egv_241`

English name: Fractional cover of Crop-, Fallow- Land within the 0.5 km landscape

Latvian name: Aramzemju, papuvju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.242 FarmlandPloughed_CropsFallow_r1250

filename: `FarmlandPloughed_CropsFallow_r1250.tif`

layername: `egv_242`

English name: Fractional cover of Crop-, Fallow- Land within the 1.25 km landscape

Latvian name: Aramzemju, papuvju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.243 FarmlandPloughed_CropsFallow_r3000

filename: `FarmlandPloughed_CropsFallow_r3000.tif`

layername: `egv_243`

English name: Fractional cover of Crop-, Fallow- Land within the 3 km landscape

Latvian name: Aramzemju, papuvju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.244 FarmlandPloughed_CropsFallow_r10000

filename: `FarmlandPloughed_CropsFallow_r10000.tif`

layername: `egv_244`

English name: Fractional cover of Crop-, Fallow- Land within the 10 km landscape

Latvian name: Aramzemju, papuvju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.245 FarmlandPloughed_CropsFallowTempGrass_cell

filename: `FarmlandPloughed_CropsFallowTempGrass_cell.tif`

layername: `egv_245`

English name: Fractional cover of Crop-, Fallow-, Temporary Grass- Lands within the analysis cell (1 ha)

Latvian name: Aramzemju, papuvju, zālāju-aramzemē platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.246 FarmlandPloughed_CropsFallowTempGrass_r500

filename: `FarmlandPloughed_CropsFallowTempGrass_r500.tif`

layername: `egv_246`

English name: Fractional cover of Crop-, Fallow-, Temporary Grass- Lands within the 0.5 km landscape

Latvian name: Aramzemju, papuvju, zālāju-aramzemē platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.247 FarmlandPloughed_CropsFallowTempGrass_r1250

filename: `FarmlandPloughed_CropsFallowTempGrass_r1250.tif`

layername: `egv_247`

English name: Fractional cover of Crop-, Fallow-, Temporary Grass- Lands within the 1.25 km landscape

Latvian name: Aramzemju, papuvju, zālāju-aramzemē platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.248 FarmlandPloughed_CropsFallowTempGrass_r3000

filename: `FarmlandPloughed_CropsFallowTempGrass_r3000.tif`

layername: `egv_248`

English name: Fractional cover of Crop-, Fallow-, Temporary Grass- Lands within the 3 km landscape

Latvian name: Aramzemju, papuvju, zālāju-aramzemē platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.249 FarmlandPloughed_CropsFallowTempGrass_r10000

filename: `FarmlandPloughed_CropsFallowTempGrass_r10000.tif`

layername: `egv_249`

English name: Fractional cover of Crop-, Fallow-, Temporary Grass- Lands within the 10 km landscape

Latvian name: Aramzemju, papuvju, zālāju-aramzemē platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.250 FarmlandPloughed_Fallow_cell

filename: `FarmlandPloughed_Fallow_cell.tif`

layername: `egv_250`

English name: Fractional cover of Fallow Land within the analysis cell (1 ha)

Latvian name: Papuvju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.251 FarmlandPloughed_Fallow_r500

filename: `FarmlandPloughed_Fallow_r500.tif`

layername: `egv_251`

English name: Fractional cover of Fallow Land within the 0.5 km landscape

Latvian name: Papuvju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.252 FarmlandPloughed_Fallow_r1250

filename: `FarmlandPloughed_Fallow_r1250.tif`

layername: `egv_252`

English name: Fractional cover of Fallow Land within the 1.25 km landscape

Latvian name: Papuvju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.253 FarmlandPloughed_Fallow_r3000

filename: `FarmlandPloughed_Fallow_r3000.tif`

layername: `egv_253`

English name: Fractional cover of Fallow Land within the 3 km landscape

Latvian name: Papuvju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.254 FarmlandPloughed_Fallow_r10000

filename: `FarmlandPloughed_Fallow_r10000.tif`

layername: `egv_254`

English name: Fractional cover of Fallow Land within the 10 km landscape

Latvian name: Papuvju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.255 FarmlandSubsidies_BiologicalSubsidies_cell

filename: `FarmlandSubsidies_BiologicalSubsidies_cell.tif`

layername: `egv_255`

English name: Fractional cover of Farmland receiving Subsidies for Biological Agriculture within the analysis cell (1 ha)

Latvian name: Bioloģiskās lauksaimniecības atbalstam pieteikto lauksaimniecības platību īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.256 FarmlandSubsidies_BiologicalSubsidies_r500

filename: `FarmlandSubsidies_BiologicalSubsidies_r500.tif`

layername: `egv_256`

English name: Fractional cover of Farmland receiving Subsidies for Biological Agriculture within the 0.5 km landscape

Latvian name: Bioloģiskās lauksaimniecības atbalstam pieteikto lauksaimniecības platību īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.257 FarmlandSubsidies_BiologicalSubsidies_r1250

filename: `FarmlandSubsidies_BiologicalSubsidies_r1250.tif`

layername: `egv_257`

English name: Fractional cover of Farmland receiving Subsidies for Biological Agriculture within the 1.25 km landscape

Latvian name: Bioloģiskās lauksaimniecības atbalstam pieteikto lauksaimniecības platību īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.258 FarmlandSubsidies_BiologicalSubsidies_r3000

filename: `FarmlandSubsidies_BiologicalSubsidies_r3000.tif`

layername: `egv_258`

English name: Fractional cover of Farmland receiving Subsidies for Biological Agriculture within the 3 km landscape

Latvian name: Bioloģiskās lauksaimniecības atbalstam pieteikto lauksaimniecības platību īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.259 FarmlandSubsidies_BiologicalSubsidies_r10000

filename: `FarmlandSubsidies_BiologicalSubsidies_r10000.tif`

layername: `egv_259`

English name: Fractional cover of Farmland receiving Subsidies for Biological Agriculture within the 10 km landscape

Latvian name: Bioloģiskās lauksaimniecības atbalstam pieteikto lauksaimniecības platību īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.260 FarmlandTrees_PermanentCrops_cell

filename: `FarmlandTrees_PermanentCrops_cell.tif`

layername: `egv_260`

English name: Fractional cover of Permanent Crops within the analysis cell (1 ha)

Latvian name: Augļudārzu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.261 FarmlandTrees_PermanentCrops_r500

filename: `FarmlandTrees_PermanentCrops_r500.tif`

layername: `egv_261`

English name: Fractional cover of Permanent Crops within the 0.5 km landscape

Latvian name: Augļudārzu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.262 FarmlandTrees_PermanentCrops_r1250

filename: `FarmlandTrees_PermanentCrops_r1250.tif`

layername: `egv_262`

English name: Fractional cover of Permanent Crops within the 1.25 km landscape

Latvian name: Augļudārzu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.263 FarmlandTrees_PermanentCrops_r3000

filename: `FarmlandTrees_PermanentCrops_r3000.tif`

layername: `egv_263`

English name: Fractional cover of Permanent Crops within the 3 km landscape

Latvian name: Augļudārzu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.264 FarmlandTrees_PermanentCrops_r10000

filename: `FarmlandTrees_PermanentCrops_r10000.tif`

layername: `egv_264`

English name: Fractional cover of Permanent Crops within the 10 km landscape

Latvian name: Augļudārzu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.265 FarmlandTrees_ShortRotationCoppice_cell

filename: `FarmlandTrees_ShortRotationCoppice_cell.tif`

layername: `egv_265`

English name: Fractional cover of Short-rotation Coppice and Other Woody Energy Crops within the analysis cell (1 ha)

Latvian name: Īscirtmeta atvasāju un enerģijai audzētu kokaugu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.266 FarmlandTrees_ShortRotationCoppice_r500

filename: `FarmlandTrees_ShortRotationCoppice_r500.tif`

layername: `egv_266`

English name: Fractional cover of Short-rotation Coppice and Other Woody Energy Crops within the 0.5 km landscape

Latvian name: Īscirtmeta atvasāju un enerģijai audzētu kokaugu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.267 FarmlandTrees_ShortRotationCoppice_r1250

filename: `FarmlandTrees_ShortRotationCoppice_r1250.tif`

layername: `egv_267`

English name: Fractional cover of Short-rotation Coppice and Other Woody Energy Crops within the 1.25 km landscape

Latvian name: Īscirtmeta atvasāju un enerģijai audzētu kokaugu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.268 FarmlandTrees_ShortRotationCoppice_r3000

filename: `FarmlandTrees_ShortRotationCoppice_r3000.tif`

layername: `egv_268`

English name: Fractional cover of Short-rotation Coppice and Other Woody Energy Crops within the 3 km landscape

Latvian name: Īscirtmeta atvasāju un enerģijai audzētu kokaugu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.269 FarmlandTrees_ShortRotationCoppice_r10000

filename: `FarmlandTrees_ShortRotationCoppice_r10000.tif`

layername: `egv_269`

English name: Fractional cover of Short-rotation Coppice and Other Woody Energy Crops within the 10 km landscape

Latvian name: Īscirtmeta atvasāju un enerģijai audzētu kokaugu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.270 ForestsAge_ClearcutsLowStands_cell

filename: `ForestsAge_ClearcutsLowStands_cell.tif`

layername: `egv_270`

English name: Fractional cover of Clearcuts and Stands lower than 5 m within the analysis cell (1 ha)

Latvian name: Izcirtumu un mežaudžu līdz 5 m augstumam platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.271 ForestsAge_ClearcutsLowStands_r500

filename: `ForestsAge_ClearcutsLowStands_r500.tif`

layername: `egv_271`

English name: Fractional cover of Clearcuts and Stands lower than 5 m within the 0.5 km landscape

Latvian name: Izcirtumu un mežaudžu līdz 5 m augstumam platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.272 ForestsAge_ClearcutsLowStands_r1250

filename: `ForestsAge_ClearcutsLowStands_r1250.tif`

layername: `egv_272`

English name: Fractional cover of Clearcuts and Stands lower than 5 m within the 1.25 km landscape

Latvian name: Izcirtumu un mežaudžu līdz 5 m augstumam platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.273 ForestsAge_ClearcutsLowStands_r3000

filename: `ForestsAge_ClearcutsLowStands_r3000.tif`

layername: `egv_273`

English name: Fractional cover of Clearcuts and Stands lower than 5 m within the 3 km landscape

Latvian name: Izcirtumu un mežaudžu līdz 5 m augstumam platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.274 ForestsAge_ClearcutsLowStands_r10000

filename: `ForestsAge_ClearcutsLowStands_r10000.tif`

layername: `egv_274`

English name: Fractional cover of Clearcuts and Stands lower than 5 m within the 10 km landscape

Latvian name: Izcirtumu un mežaudžu līdz 5 m augstumam platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.275 ForestsAge_Middle_cell

filename: `ForestsAge_Middle_cell.tif`

layername: `egv_275`

English name: Fractional cover of Middle-Aged Forests within the analysis cell (1 ha)

Latvian name: Vidēja vecuma un briestaudžu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.276 ForestsAge_Middle_r500

filename: `ForestsAge_Middle_r500.tif`

layername: `egv_276`

English name: Fractional cover of Middle-Aged Forests within the 0.5 km landscape

Latvian name: Vidēja vecuma un briestaudžu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.277 ForestsAge_Middle_r1250

filename: `ForestsAge_Middle_r1250.tif`

layername: `egv_277`

English name: Fractional cover of Middle-Aged Forests within the 1.25 km landscape

Latvian name: Vidēja vecuma un briestaudžu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.278 ForestsAge_Middle_r3000

filename: `ForestsAge_Middle_r3000.tif`

layername: `egv_278`

English name: Fractional cover of Middle-Aged Forests within the 3 km landscape

Latvian name: Vidēja vecuma un briestaudžu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.279 ForestsAge_Middle_r10000

filename: `ForestsAge_Middle_r10000.tif`

layername: `egv_279`

English name: Fractional cover of Middle-Aged Forests within the 10 km landscape

Latvian name: Vidēja vecuma un briestaudžu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.280 ForestsAge_Old_cell

filename: `ForestsAge_Old_cell.tif`

layername: `egv_280`

English name: Fractional cover of Old (over rotation age) Forests within the analysis cell (1 ha)

Latvian name: Vecu (kopš cirtmeta) mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.281 ForestsAge_Old_r500

filename: `ForestsAge_Old_r500.tif`

layername: `egv_281`

English name: Fractional cover of Old (over rotation age) Forests within the 0.5 km landscape

Latvian name: Vecu (kopš cirtmeta)mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.282 ForestsAge_Old_r1250

filename: `ForestsAge_Old_r1250.tif`

layername: `egv_282`

English name: Fractional cover of Old (over rotation age) Forests within the 1.25 km landscape

Latvian name: Vecu (kopš cirtmeta) mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.283 ForestsAge_Old_r3000

filename: `ForestsAge_Old_r3000.tif`

layername: `egv_283`

English name: Fractional cover of Old (over rotation age) Forests within the 3 km landscape

Latvian name: Vecu (kopš cirtmeta) mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.284 ForestsAge_Old_r10000

filename: `ForestsAge_Old_r10000.tif`

layername: `egv_284`

English name: Fractional cover of Old (over rotation age) Forests within the 10 km landscape

Latvian name: Vecu (kopš cirtmeta) mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.285 ForestsAge_YoungTallStandsShrubs_cell

filename: `ForestsAge_YoungTallStandsShrubs_cell.tif`

layername: `egv_285`

English name: Fractional cover of Shrubs, Young Stands (at least 5 m tall) within the analysis cell (1 ha)

Latvian name: Krūmāju un jaunaudžu (no 5 m augstuma) platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.286 ForestsAge_YoungTallStandsShrubs_r500

filename: `ForestsAge_YoungTallStandsShrubs_r500.tif`

layername: `egv_286`

English name: Fractional cover of Shrubs, Young Stands (at least 5 m tall) within the 0.5 km landscape

Latvian name: Krūmāju un jaunaudžu (no 5 m augstuma) platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.287 ForestsAge_YoungTallStandsShrubs_r1250

filename: `ForestsAge_YoungTallStandsShrubs_r1250.tif`

layername: `egv_287`

English name: Fractional cover of Shrubs, Young Stands (at least 5 m tall) within the 1.25 km landscape

Latvian name: Krūmāju un jaunaudžu (no 5 m augstuma) platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.288 ForestsAge_YoungTallStandsShrubs_r3000

filename: `ForestsAge_YoungTallStandsShrubs_r3000.tif`

layername: `egv_288`

English name: Fractional cover of Shrubs, Young Stands (at least 5 m tall) within the 3 km landscape

Latvian name: Krūmāju un jaunaudžu (no 5 m augstuma) platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.289 ForestsAge_YoungTallStandsShrubs_r10000

filename: `ForestsAge_YoungTallStandsShrubs_r10000.tif`

layername: `egv_289`

English name: Fractional cover of Shrubs, Young Stands (at least 5 m tall) within the 10 km landscape

Latvian name: Krūmāju un jaunaudžu (no 5 m augstuma) platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.290 ForestsQuant_AgeProp-average_cell

filename: `ForestsQuant_AgeProp-average_cell.tif`

layername: `egv_290`

English name: Average stand age relative to rotation age within the analysis cell (1 ha)

Latvian name: Mežaudzes vecuma attiecība pret cirtmetu, vidējais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.291 ForestsQuant_DominantDiameter-max_cell

filename: `ForestsQuant_DominantDiameter-max_cell.tif`

layername: `egv_291`

English name: Dominant tree trunk diameter, maximum within the analysis cell (1 ha)

Latvian name: Koku stumbra diametrs, valdaudzes maksimālais analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.292 ForestsQuant_LargestDiameter-max_cell

filename: `ForestsQuant_LargestDiameter-max_cell.tif`

layername: `egv_292`

English name: Largest tree trunk diameter within the analysis cell (1 ha)

Latvian name: Lielākais koka stumbra diametrs analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.293 ForestsQuant_TimeSinceDisturbance-average_cell

filename: `ForestsQuant_TimeSinceDisturbance-average_cell.tif`

layername: `egv_293`

English name: Time since last disturbance affecting tree growing within the analysis cell (1 ha)

Latvian name: Laiks kopš pēdējā ar koku augšanu saistītā traucējuma analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.294 ForestsQuant_VolumeAspen-sum_cell

filename: `ForestsQuant_VolumeAspen-sum_cell.tif`

layername: `egv_294`

English name: Timber volume of Aspens, Poplars within the analysis cell (1 ha)

Latvian name: Apšu, papeļu krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.295 ForestsQuant_VolumeBirch-sum_cell

filename: `ForestsQuant_VolumeBirch-sum_cell.tif`

layername: `egv_295`

English name: Timber volume of Birches within the analysis cell (1 ha)

Latvian name: Bērzu krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.296 ForestsQuant_VolumeBlackAlder-sum_cell

filename: `ForestsQuant_VolumeBlackAlder-sum_cell.tif`

layername: `egv_296`

English name: Timber volume of Black Alder within the analysis cell (1 ha)

Latvian name: Melnalkšņu krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.297 ForestsQuant_VolumeBorealDeciduousOther-sum_cell

filename: `ForestsQuant_VolumeBorealDeciduousOther-sum_cell.tif`

layername: `egv_297`

English name: Timber volume of Other Boreal Deciduous trees within the analysis cell (1 ha)

Latvian name: Citu šaurlapju krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.298 ForestsQuant_VolumeBorealDeciduousTotal-sum_cell

filename: `ForestsQuant_VolumeBorealDeciduousTotal-sum_cell.tif`

layername: `egv_298`

English name: Timber volume of Boreal Deciduous trees within the analysis cell (1 ha)

Latvian name: Šaurlapju krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.299 ForestsQuant_VolumeConiferous-sum_cell

filename: `ForestsQuant_VolumeConiferous-sum_cell.tif`

layername: `egv_299`

English name: Timber volume of Coniferous trees within the analysis cell (1 ha)

Latvian name: Skujkoku krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.300 ForestsQuant_VolumeOak-sum_cell

filename: `ForestsQuant_VolumeOak-sum_cell.tif`

layername: `egv_300`

English name: Timber volume of Oaks within the analysis cell (1 ha)

Latvian name: Ozolu krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.301 ForestsQuant_VolumeOakMaple-sum_cell

filename: `ForestsQuant_VolumeOakMaple-sum_cell.tif`

layername: `egv_301`

English name: Timber volume of Oaks, Maples within the analysis cell (1 ha)

Latvian name: Ozolu, kļavu krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.302 ForestsQuant_VolumePine-sum_cell

filename: `ForestsQuant_VolumePine-sum_cell.tif`

layername: `egv_302`

English name: Timber volume of Pines within the analysis cell (1 ha)

Latvian name: Priežu krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.303 ForestsQuant_VolumeSpruce-sum_cell

filename: `ForestsQuant_VolumeSpruce-sum_cell.tif`

layername: `egv_303`

English name: Timber volume of Spruces within the analysis cell (1 ha)

Latvian name: Egļu krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.304 ForestsQuant_VolumeTemperateDeciduousTotal-sum_cell

filename: `ForestsQuant_VolumeTemperateDeciduousTotal-sum_cell.tif`

layername: `egv_304`

English name: Timber volume of Temperate Deciduous trees within the analysis cell (1 ha)

Latvian name: Platlapju krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.305 ForestsQuant_VolumeTemperateWithoutOak-sum_cell

filename: `ForestsQuant_VolumeTemperateWithoutOak-sum_cell.tif`

layername: `egv_305`

English name: Timber volume of Temperate Deciduous trees (without oaks) within the analysis cell (1 ha)

Latvian name: Paltlapju (bez ozoliem) krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.306 ForestsQuant_VolumeTemperateWithoutOakMaple-sum_cell

filename: `ForestsQuant_VolumeTemperateWithoutOakMaple-sum_cell.tif`

layername: `egv_306`

English name: Timber volume of Temperate Deciduous trees (without oaks, maples) within the analysis cell (1 ha)

Latvian name: Platlapju (bez ozoliem, kļavām) krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.307 ForestsQuant_VolumeTotal-sum_cell

filename: `ForestsQuant_VolumeTotal-sum_cell.tif`

layername: `egv_307`

English name: Timber volume within the analysis cell (1 ha)

Latvian name: Kopējā krāja analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.308 ForestsSoil_EutrophicDrained_cell

filename: `ForestsSoil_EutrophicDrained_cell.tif`

layername: `egv_308`

English name: Fractional cover of Drained Eutrophic Forests within the analysis cell (1 ha)

Latvian name: Susinātu eitrofu mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.309 ForestsSoil_EutrophicDrained_r500

filename: `ForestsSoil_EutrophicDrained_r500.tif`

layername: `egv_309`

English name: Fractional cover of Drained Eutrophic Forests within the 0.5 km landscape

Latvian name: Susinātu eitrofu mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.310 ForestsSoil_EutrophicDrained_r1250

filename: `ForestsSoil_EutrophicDrained_r1250.tif`

layername: `egv_310`

English name: Fractional cover of Drained Eutrophic Forests within the 1.25 km landscape

Latvian name: Susinātu eitrofu mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.311 ForestsSoil_EutrophicDrained_r3000

filename: `ForestsSoil_EutrophicDrained_r3000.tif`

layername: `egv_311`

English name: Fractional cover of Drained Eutrophic Forests within the 3 km landscape

Latvian name: Susinātu eitrofu mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.312 ForestsSoil_EutrophicDrained_r10000

filename: `ForestsSoil_EutrophicDrained_r10000.tif`

layername: `egv_312`

English name: Fractional cover of Drained Eutrophic Forests within the 10 km landscape

Latvian name: Susinātu eitrofu mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.313 ForestsSoil_EutrophicMineral_cell

filename: `ForestsSoil_EutrophicMineral_cell.tif`

layername: `egv_313`

English name: Fractional cover of Eutrophic Forests on undrained Mineral Soils within the analysis cell (1 ha)

Latvian name: Eitrofu mežu nesusinātās minerālaugsnēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.314 ForestsSoil_EutrophicMineral_r500

filename: `ForestsSoil_EutrophicMineral_r500.tif`

layername: `egv_314`

English name: Fractional cover of Eutrophic Forests on undrained Mineral Soils within the 0.5 km landscape

Latvian name: Eitrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.315 ForestsSoil_EutrophicMineral_r1250

filename: `ForestsSoil_EutrophicMineral_r1250.tif`

layername: `egv_315`

English name: Fractional cover of Eutrophic Forests on undrained Mineral Soils within the 1.25 km landscape

Latvian name: Eitrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.316 ForestsSoil_EutrophicMineral_r3000

filename: `ForestsSoil_EutrophicMineral_r3000.tif`

layername: `egv_316`

English name: Fractional cover of Eutrophic Forests on undrained Mineral Soils within the 3 km landscape

Latvian name: Eitrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.317 ForestsSoil_EutrophicMineral_r10000

filename: `ForestsSoil_EutrophicMineral_r10000.tif`

layername: `egv_317`

English name: Fractional cover of Eutrophic Forests on undrained Mineral Soils within the 10 km landscape

Latvian name: Eitrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.318 ForestsSoil_EutrophicOrganic_cell

filename: `ForestsSoil_EutrophicOrganic_cell.tif`

layername: `egv_318`

English name: Fractional cover of Eutrophic Forests on undrained Organic Soils within the analysis cell (1 ha)

Latvian name: Eitrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.319 ForestsSoil_EutrophicOrganic_r500

filename: `ForestsSoil_EutrophicOrganic_r500.tif`

layername: `egv_319`

English name: Fractional cover of Eutrophic Forests on undrained Organic Soils within the 0.5 km landscape

Latvian name: Eitrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.320 ForestsSoil_EutrophicOrganic_r1250

filename: `ForestsSoil_EutrophicOrganic_r1250.tif`

layername: `egv_320`

English name: Fractional cover of Eutrophic Forests on undrained Organic Soils within the 1.25 km landscape

Latvian name: Eitrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.321 ForestsSoil_EutrophicOrganic_r3000

filename: `ForestsSoil_EutrophicOrganic_r3000.tif`

layername: `egv_321`

English name: Fractional cover of Eutrophic Forests on undrained Organic Soils within the 3 km landscape

Latvian name: Eitrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.322 ForestsSoil_EutrophicOrganic_r10000

filename: `ForestsSoil_EutrophicOrganic_r10000.tif`

layername: `egv_322`

English name: Fractional cover of Eutrophic Forests on undrained Organic Soils within the 10 km landscape

Latvian name: Eitrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.323 ForestsSoil_MesotrophicMineral_cell

filename: `ForestsSoil_MesotrophicMineral_cell.tif`

layername: `egv_323`

English name: Fractional cover of Mesotrophic Forests on undrained Mineral Soils within the analysis cell (1 ha)

Latvian name: Mezotrofu mežu minerālaugsnēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.324 ForestsSoil_MesotrophicMineral_r500

filename: `ForestsSoil_MesotrophicMineral_r500.tif`

layername: `egv_324`

English name: Fractional cover of Mesotrophic Forests on undrained Mineral Soils within the 0.5 km landscape

Latvian name: Mezotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.325 ForestsSoil_MesotrophicMineral_r1250

filename: `ForestsSoil_MesotrophicMineral_r1250.tif`

layername: `egv_325`

English name: Fractional cover of Mesotrophic Forests on undrained Mineral Soils within the 1.25 km landscape

Latvian name: Mezotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.326 ForestsSoil_MesotrophicMineral_r3000

filename: `ForestsSoil_MesotrophicMineral_r3000.tif`

layername: `egv_326`

English name: Fractional cover of Mesotrophic Forests on undrained Mineral Soils within the 3 km landscape

Latvian name: Mezotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.327 ForestsSoil_MesotrophicMineral_r10000

filename: `ForestsSoil_MesotrophicMineral_r10000.tif`

layername: `egv_327`

English name: Fractional cover of Mesotrophic Forests on undrained Mineral Soils within the 10 km landscape

Latvian name: Mezotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.328 ForestsSoil_OligotrophicDrained_cell

filename: `ForestsSoil_OligotrophicDrained_cell.tif`

layername: `egv_328`

English name: Fractional cover of Drained Oligotrophic Forests within the analysis cell (1 ha)

Latvian name: Susinātu oligotrofu mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.329 ForestsSoil_OligotrophicDrained_r500

filename: `ForestsSoil_OligotrophicDrained_r500.tif`

layername: `egv_329`

English name: Fractional cover of Drained Oligotrophic Forests within the 0.5 km landscape

Latvian name: Susinātu oligotrofu mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.330 ForestsSoil_OligotrophicDrained_r1250

filename: `ForestsSoil_OligotrophicDrained_r1250.tif`

layername: `egv_330`

English name: Fractional cover of Drained Oligotrophic Forests within the 1.25 km landscape

Latvian name: Susinātu oligotrofu mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.331 ForestsSoil_OligotrophicDrained_r3000

filename: `ForestsSoil_OligotrophicDrained_r3000.tif`

layername: `egv_331`

English name: Fractional cover of Drained Oligotrophic Forests within the 3 km landscape

Latvian name: Susinātu oligotrofu mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.332 ForestsSoil_OligotrophicDrained_r10000

filename: `ForestsSoil_OligotrophicDrained_r10000.tif`

layername: `egv_332`

English name: Fractional cover of Drained Oligotrophic Forests within the 10 km landscape

Latvian name: Susinātu oligotrofu mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.333 ForestsSoil_OligotrophicMineral_cell

filename: `ForestsSoil_OligotrophicMineral_cell.tif`

layername: `egv_333`

English name: Fractional cover of Oligotrophic Forests on undrained Mineral Soils within the analysis cell (1 ha)

Latvian name: Oligotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.334 ForestsSoil_OligotrophicMineral_r500

filename: `ForestsSoil_OligotrophicMineral_r500.tif`

layername: `egv_334`

English name: Fractional cover of Oligotrophic Forests on undrained Mineral Soils within the 0.5 km landscape

Latvian name: Oligotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.335 ForestsSoil_OligotrophicMineral_r1250

filename: `ForestsSoil_OligotrophicMineral_r1250.tif`

layername: `egv_335`

English name: Fractional cover of Oligotrophic Forests on undrained Mineral Soils within the 1.25 km landscape

Latvian name: Oligotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.336 ForestsSoil_OligotrophicMineral_r3000

filename: `ForestsSoil_OligotrophicMineral_r3000.tif`

layername: `egv_336`

English name: Fractional cover of Oligotrophic Forests on undrained Mineral Soils within the 3 km landscape

Latvian name: Oligotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.337 ForestsSoil_OligotrophicMineral_r10000

filename: `ForestsSoil_OligotrophicMineral_r10000.tif`

layername: `egv_337`

English name: Fractional cover of Oligotrophic Forests on undrained Mineral Soils within the 10 km landscape

Latvian name: Oligotrofu mežu nesusinātās minerālaugsnēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.338 ForestsSoil_OligotrophicOrganic_cell

filename: `ForestsSoil_OligotrophicOrganic_cell.tif`

layername: `egv_338`

English name: Fractional cover of Oligotrophic Forests on undrained Organic Soils within the analysis cell (1 ha)

Latvian name: Oligotrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.339 ForestsSoil_OligotrophicOrganic_r500

filename: `ForestsSoil_OligotrophicOrganic_r500.tif`

layername: `egv_339`

English name: Fractional cover of Oligotrophic Forests on undrained Organic Soils within the 0.5 km landscape

Latvian name: Oligotrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.340 ForestsSoil_OligotrophicOrganic_r1250

filename: `ForestsSoil_OligotrophicOrganic_r1250.tif`

layername: `egv_340`

English name: Fractional cover of Oligotrophic Forests on undrained Organic Soils within the 1.25 km landscape

Latvian name: Oligotrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.341 ForestsSoil_OligotrophicOrganic_r3000

filename: `ForestsSoil_OligotrophicOrganic_r3000.tif`

layername: `egv_341`

English name: Fractional cover of Oligotrophic Forests on undrained Organic Soils within the 3 km landscape

Latvian name: Oligotrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.342 ForestsSoil_OligotrophicOrganic_r10000

filename: `ForestsSoil_OligotrophicOrganic_r10000.tif`

layername: `egv_342`

English name: Fractional cover of Oligotrophic Forests on undrained Organic Soils within the 10 km landscape

Latvian name: Oligotrofu mežu nesusinātās organiskajās augsnēs platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.343 ForestsTreesAge_BorealDeciduousOld_cell

filename: `ForestsTreesAge_BorealDeciduousOld_cell.tif`

layername: `egv_343`

English name: Fractional cover of Old (over rotation age) Boreal Deciduous Forests within the analysis cell (1 ha)

Latvian name: Vecu (kopš cirtmeta) šaurlapju mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.344 ForestsTreesAge_BorealDeciduousOld_r500

filename: `ForestsTreesAge_BorealDeciduousOld_r500.tif`

layername: `egv_344`

English name: Fractional cover of Old (over rotation age) Boreal Deciduous Forests within the 0.5 km landscape

Latvian name: Vecu (kopš cirtmeta) šaurlapju mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.345 ForestsTreesAge_BorealDeciduousOld_r1250

filename: `ForestsTreesAge_BorealDeciduousOld_r1250.tif`

layername: `egv_345`

English name: Fractional cover of Old (over rotation age) Boreal Deciduous Forests within the 1.25 km landscape

Latvian name: Vecu (kopš cirtmeta) šaurlapju mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.346 ForestsTreesAge_BorealDeciduousOld_r3000

filename: `ForestsTreesAge_BorealDeciduousOld_r3000.tif`

layername: `egv_346`

English name: Fractional cover of Old (over rotation age) Boreal Deciduous Forests within the 3 km landscape

Latvian name: Vecu (kopš cirtmeta) šaurlapju mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.347 ForestsTreesAge_BorealDeciduousOld_r10000

filename: `ForestsTreesAge_BorealDeciduousOld_r10000.tif`

layername: `egv_347`

English name: Fractional cover of Old (over rotation age) Boreal Deciduous Forests within the 10 km landscape

Latvian name: Vecu (kopš cirtmeta) šaurlapju mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.348 ForestsTreesAge_BorealDeciduousYoung_cell

filename: `ForestsTreesAge_BorealDeciduousYoung_cell.tif`

layername: `egv_348`

English name: Fractional cover of Young (pre-rotation age) Boreal Deciduous Forests within the analysis cell (1 ha)

Latvian name: Jaunu (pirms cirtmeta) šaurlapju mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.349 ForestsTreesAge_BorealDeciduousYoung_r500

filename: `ForestsTreesAge_BorealDeciduousYoung_r500.tif`

layername: `egv_349`

English name: Fractional cover of Young (pre-rotation age) Boreal Deciduous Forests within the 0.5 km landscape

Latvian name: Jaunu (pirms cirtmeta) šaurlapju mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.350 ForestsTreesAge_BorealDeciduousYoung_r1250

filename: `ForestsTreesAge_BorealDeciduousYoung_r1250.tif`

layername: `egv_350`

English name: Fractional cover of Young (pre-rotation age) Boreal Deciduous Forests within the 1.25 km landscape

Latvian name: Jaunu (pirms cirtmeta) šaurlapju mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.351 ForestsTreesAge_BorealDeciduousYoung_r3000

filename: `ForestsTreesAge_BorealDeciduousYoung_r3000.tif`

layername: `egv_351`

English name: Fractional cover of Young (pre-rotation age) Boreal Deciduous Forests within the 3 km landscape

Latvian name: Jaunu (pirms cirtmeta) šaurlapju mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.352 ForestsTreesAge_BorealDeciduousYoung_r10000

filename: `ForestsTreesAge_BorealDeciduousYoung_r10000.tif`

layername: `egv_352`

English name: Fractional cover of Young (pre-rotation age) Boreal Deciduous Forests within the 10 km landscape

Latvian name: Jaunu (pirms cirtmeta) šaurlapju mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.353 ForestsTreesAge_ConiferousOld_cell

filename: `ForestsTreesAge_ConiferousOld_cell.tif`

layername: `egv_353`

English name: Fractional cover of Old (over rotation age) Coniferous Forests within the analysis cell (1 ha)

Latvian name: Vecu (kopš cirtmeta) skujkoku mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.354 ForestsTreesAge_ConiferousOld_r500

filename: `ForestsTreesAge_ConiferousOld_r500.tif`

layername: `egv_354`

English name: Fractional cover of Old (over rotation age) Coniferous Forests within the 0.5 km landscape

Latvian name: Vecu (kopš cirtmeta) skujkoku mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.355 ForestsTreesAge_ConiferousOld_r1250

filename: `ForestsTreesAge_ConiferousOld_r1250.tif`

layername: `egv_355`

English name: Fractional cover of Old (over rotation age) Coniferous Forests within the 1.25 km landscape

Latvian name: Vecu (kopš cirtmeta) skujkoku mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.356 ForestsTreesAge_ConiferousOld_r3000

filename: `ForestsTreesAge_ConiferousOld_r3000.tif`

layername: `egv_356`

English name: Fractional cover of Old (over rotation age) Coniferous Forests within the 3 km landscape

Latvian name: Vecu (kopš cirtmeta) skujkoku mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.357 ForestsTreesAge_ConiferousOld_r10000

filename: `ForestsTreesAge_ConiferousOld_r10000.tif`

layername: `egv_357`

English name: Fractional cover of Old (over rotation age) Coniferous Forests within the 10 km landscape

Latvian name: Vecu (kopš cirtmeta) skujkoku mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.358 ForestsTreesAge_ConiferousYoung_cell

filename: `ForestsTreesAge_ConiferousYoung_cell.tif`

layername: `egv_358`

English name: Fractional cover of Young (pre-rotation age) Coniferous Forests within the analysis cell (1 ha)

Latvian name: Jaunu (pirms cirtmeta) skujkoku mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.359 ForestsTreesAge_ConiferousYoung_r500

filename: `ForestsTreesAge_ConiferousYoung_r500.tif`

layername: `egv_359`

English name: Fractional cover of Young (pre-rotation age) Coniferous Forests within the 0.5 km landscape

Latvian name: Jaunu (pirms cirtmeta) skujkoku mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.360 ForestsTreesAge_ConiferousYoung_r1250

filename: `ForestsTreesAge_ConiferousYoung_r1250.tif`

layername: `egv_360`

English name: Fractional cover of Young (pre-rotation age) Coniferous Forests within the 1.25 km landscape

Latvian name: Jaunu (pirms cirtmeta) skujkoku mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.361 ForestsTreesAge_ConiferousYoung_r3000

filename: `ForestsTreesAge_ConiferousYoung_r3000.tif`

layername: `egv_361`

English name: Fractional cover of Young (pre-rotation age) Coniferous Forests within the 3 km landscape

Latvian name: Jaunu (pirms cirtmeta) skujkoku mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.362 ForestsTreesAge_ConiferousYoung_r10000

filename: `ForestsTreesAge_ConiferousYoung_r10000.tif`

layername: `egv_362`

English name: Fractional cover of Young (pre-rotation age) Coniferous Forests within the 10 km landscape

Latvian name: Jaunu (pirms cirtmeta) skujkoku mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.363 ForestsTreesAge_MixedOld_cell

filename: `ForestsTreesAge_MixedOld_cell.tif`

layername: `egv_363`

English name: Fractional cover of Old (over rotation age) Mixed Forests within the analysis cell (1 ha)

Latvian name: Vecu (kopš cirtmeta) jauktu koku mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.364 ForestsTreesAge_MixedOld_r500

filename: `ForestsTreesAge_MixedOld_r500.tif`

layername: `egv_364`

English name: Fractional cover of Old (over rotation age) Mixed Forests within the 0.5 km landscape

Latvian name: Vecu (kopš cirtmeta) jauktu koku mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.365 ForestsTreesAge_MixedOld_r1250

filename: `ForestsTreesAge_MixedOld_r1250.tif`

layername: `egv_365`

English name: Fractional cover of Old (over rotation age) Mixed Forests within the 1.25 km landscape

Latvian name: Vecu (kopš cirtmeta) jauktu koku mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.366 ForestsTreesAge_MixedOld_r3000

filename: `ForestsTreesAge_MixedOld_r3000.tif`

layername: `egv_366`

English name: Fractional cover of Old (over rotation age) Mixed Forests within the 3 km landscape

Latvian name: Vecu (kopš cirtmeta) jauktu koku mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.367 ForestsTreesAge_MixedOld_r10000

filename: `ForestsTreesAge_MixedOld_r10000.tif`

layername: `egv_367`

English name: Fractional cover of Old (over rotation age) Mixed Forests within the 10 km landscape

Latvian name: Vecu (kopš cirtmeta) jauktu koku mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.368 ForestsTreesAge_MixedYoung_cell

filename: `ForestsTreesAge_MixedYoung_cell.tif`

layername: `egv_368`

English name: Fractional cover of Young (pre-rotation age) Mixed Forests within the analysis cell (1 ha)

Latvian name: Jaunu (pirms cirtmeta) jauktu koku mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.369 ForestsTreesAge_MixedYoung_r500

filename: `ForestsTreesAge_MixedYoung_r500.tif`

layername: `egv_369`

English name: Fractional cover of Young (pre-rotation age) Mixed Forests within the 0.5 km landscape

Latvian name: Jaunu (pirms cirtmeta) jauktu koku mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.370 ForestsTreesAge_MixedYoung_r1250

filename: `ForestsTreesAge_MixedYoung_r1250.tif`

layername: `egv_370`

English name: Fractional cover of Young (pre-rotation age) Mixed Forests within the 1.25 km landscape

Latvian name: Jaunu (pirms cirtmeta) jauktu koku mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.371 ForestsTreesAge_MixedYoung_r3000

filename: `ForestsTreesAge_MixedYoung_r3000.tif`

layername: `egv_371`

English name: Fractional cover of Young (pre-rotation age) Mixed Forests within the 3 km landscape

Latvian name: Jaunu (pirms cirtmeta) jauktu koku mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.372 ForestsTreesAge_MixedYoung_r10000

filename: `ForestsTreesAge_MixedYoung_r10000.tif`

layername: `egv_372`

English name: Fractional cover of Young (pre-rotation age) Mixed Forests within the 10 km landscape

Latvian name: Jaunu (pirms cirtmeta) jauktu koku mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.373 ForestsTreesAge_TemperateDeciduousOld_cell

filename: `ForestsTreesAge_TemperateDeciduousOld_cell.tif`

layername: `egv_373`

English name: Fractional cover of Old (over rotation age) Temperate Deciduous Forests within the analysis cell (1 ha)

Latvian name: Vecu (kopš cirtmeta) platlapju mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.374 ForestsTreesAge_TemperateDeciduousOld_r500

filename: `ForestsTreesAge_TemperateDeciduousOld_r500.tif`

layername: `egv_374`

English name: Fractional cover of Old (over rotation age) Temperate Deciduous Forests within the 0.5 km landscape

Latvian name: Vecu (kopš cirtmeta) platlapju mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.375 ForestsTreesAge_TemperateDeciduousOld_r1250

filename: `ForestsTreesAge_TemperateDeciduousOld_r1250.tif`

layername: `egv_375`

English name: Fractional cover of Old (over rotation age) Temperate Deciduous Forests within the 1.25 km landscape

Latvian name: Vecu (kopš cirtmeta) platlapju mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.376 ForestsTreesAge_TemperateDeciduousOld_r3000

filename: `ForestsTreesAge_TemperateDeciduousOld_r3000.tif`

layername: `egv_376`

English name: Fractional cover of Old (over rotation age) Temperate Deciduous Forests within the 3 km landscape

Latvian name: Vecu (kopš cirtmeta) platlapju mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.377 ForestsTreesAge_TemperateDeciduousOld_r10000

filename: `ForestsTreesAge_TemperateDeciduousOld_r10000.tif`

layername: `egv_377`

English name: Fractional cover of Old (over rotation age) Temperate Deciduous Forests within the 10 km landscape

Latvian name: Vecu (kopš cirtmeta) platlapju mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.378 ForestsTreesAge_TemperateDeciduousYoung_cell

filename: `ForestsTreesAge_TemperateDeciduousYoung_cell.tif`

layername: `egv_378`

English name: Fractional cover of Young (pre-rotation age) Temperate Deciduous Forests within the analysis cell (1 ha)

Latvian name: Jaunu (pirms cirtmeta) platlapju mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.379 ForestsTreesAge_TemperateDeciduousYoung_r500

filename: `ForestsTreesAge_TemperateDeciduousYoung_r500.tif`

layername: `egv_379`

English name: Fractional cover of Young (pre-rotation age) Temperate Deciduous Forests within the 0.5 km landscape

Latvian name: Jaunu (pirms cirtmeta) platlapju mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.380 ForestsTreesAge_TemperateDeciduousYoung_r1250

filename: `ForestsTreesAge_TemperateDeciduousYoung_r1250.tif`

layername: `egv_380`

English name: Fractional cover of Young (pre-rotation age) Temperate Deciduous Forests within the 1.25 km landscape

Latvian name: Jaunu (pirms cirtmeta) platlapju mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.381 ForestsTreesAge_TemperateDeciduousYoung_r3000

filename: `ForestsTreesAge_TemperateDeciduousYoung_r3000.tif`

layername: `egv_381`

English name: Fractional cover of Young (pre-rotation age) Temperate Deciduous Forests within the 3 km landscape

Latvian name: Jaunu (pirms cirtmeta) platlapju mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.382 ForestsTreesAge_TemperateDeciduousYoung_r10000

filename: `ForestsTreesAge_TemperateDeciduousYoung_r10000.tif`

layername: `egv_382`

English name: Fractional cover of Young (pre-rotation age) Temperate Deciduous Forests within the 10 km landscape

Latvian name: Jaunu (pirms cirtmeta) platlapju mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.383 ForestsTrees_BorealDeciduous_cell

filename: `ForestsTrees_BorealDeciduous_cell.tif`

layername: `egv_383`

English name: Fractional cover of Boeral Deciduous Forests within the analysis cell (1 ha)

Latvian name: Šaurlapju mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.384 ForestsTrees_BorealDeciduous_r500

filename: `ForestsTrees_BorealDeciduous_r500.tif`

layername: `egv_384`

English name: Fractional cover of Boreal Deciduous Forests within the 0.5 km landscape

Latvian name: Šaurlapju mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.385 ForestsTrees_BorealDeciduous_r1250

filename: `ForestsTrees_BorealDeciduous_r1250.tif`

layername: `egv_385`

English name: Fractional cover of Boreal Deciduous Forests within the 1.25 km landscape

Latvian name: Šaurlapju mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.386 ForestsTrees_BorealDeciduous_r3000

filename: `ForestsTrees_BorealDeciduous_r3000.tif`

layername: `egv_386`

English name: Fractional cover of Boreal Deciduous Forests within the 3 km landscape

Latvian name: Šaurlapju mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.387 ForestsTrees_BorealDeciduous_r10000

filename: `ForestsTrees_BorealDeciduous_r10000.tif`

layername: `egv_387`

English name: Fractional cover of Boreal Deciduous Forests within the 10 km landscape

Latvian name: Šaurlapju mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.388 ForestsTrees_Coniferous_cell

filename: `ForestsTrees_Coniferous_cell.tif`

layername: `egv_388`

English name: Fractional cover of Coniferous Forests within the analysis cell (1 ha)

Latvian name: Skujkoku mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.389 ForestsTrees_Coniferous_r500

filename: `ForestsTrees_Coniferous_r500.tif`

layername: `egv_389`

English name: Fractional cover of Coniferous Forests within the 0.5 km landscape

Latvian name: Skujkoku mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.390 ForestsTrees_Coniferous_r1250

filename: `ForestsTrees_Coniferous_r1250.tif`

layername: `egv_390`

English name: Fractional cover of Coniferous Forests within the 1.25 km landscape

Latvian name: Skujkoku mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.391 ForestsTrees_Coniferous_r3000

filename: `ForestsTrees_Coniferous_r3000.tif`

layername: `egv_391`

English name: Fractional cover of Coniferous Forests within the 3 km landscape

Latvian name: Skujkoku mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.392 ForestsTrees_Coniferous_r10000

filename: `ForestsTrees_Coniferous_r10000.tif`

layername: `egv_392`

English name: Fractional cover of Coniferous Forests within the 10 km landscape

Latvian name: Skujkoku mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.393 ForestsTrees_Mixed_cell

filename: `ForestsTrees_Mixed_cell.tif`

layername: `egv_393`

English name: Fractional cover of Mixed Forests within the analysis cell (1 ha)

Latvian name: Jauktu koku mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.394 ForestsTrees_Mixed_r500

filename: `ForestsTrees_Mixed_r500.tif`

layername: `egv_394`

English name: Fractional cover of Mixed Forests within the 0.5 km landscape

Latvian name: Jauktu koku mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.395 ForestsTrees_Mixed_r1250

filename: `ForestsTrees_Mixed_r1250.tif`

layername: `egv_395`

English name: Fractional cover of Mixed Forests within the 1.25 km landscape

Latvian name: Jauktu koku mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.396 ForestsTrees_Mixed_r3000

filename: `ForestsTrees_Mixed_r3000.tif`

layername: `egv_396`

English name: Fractional cover of Mixed Forests within the 3 km landscape

Latvian name: Jauktu koku mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.397 ForestsTrees_Mixed_r10000

filename: `ForestsTrees_Mixed_r10000.tif`

layername: `egv_397`

English name: Fractional cover of Mixed Forests within the 10 km landscape

Latvian name: Jauktu koku mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.398 ForestsTrees_TemperateDeciduous_cell

filename: `ForestsTrees_TemperateDeciduous_cell.tif`

layername: `egv_398`

English name: Fractional cover of Temperate Deciduous Forests within the analysis cell (1 ha)

Latvian name: Platlapju mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.399 ForestsTrees_TemperateDeciduous_r500

filename: `ForestsTrees_TemperateDeciduous_r500.tif`

layername: `egv_399`

English name: Fractional cover of Temperate Deciduous Forests within the 0.5 km landscape

Latvian name: Platlapju mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.400 ForestsTrees_TemperateDeciduous_r1250

filename: `ForestsTrees_TemperateDeciduous_r1250.tif`

layername: `egv_400`

English name: Fractional cover of Temperate Deciduous Forests within the 1.25 km landscape

Latvian name: Platlapju mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.401 ForestsTrees_TemperateDeciduous_r3000

filename: `ForestsTrees_TemperateDeciduous_r3000.tif`

layername: `egv_401`

English name: Fractional cover of Temperate Deciduous Forests within the 3 km landscape

Latvian name: Platlapju mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.402 ForestsTrees_TemperateDeciduous_r10000

filename: `ForestsTrees_TemperateDeciduous_r10000.tif`

layername: `egv_402`

English name: Fractional cover of Temperate Deciduous Forests within the 10 km landscape

Latvian name: Platlapju mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.403 General_AllotmentGardens_cell

filename: `General_AllotmentGardens_cell.tif`

layername: `egv_403`

English name: Fractional cover of Allotment gardens within the analysis cell (1 ha)

Latvian name: Vasarnīcu kompleksu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.404 General_AllotmentGardens_r500

filename: `General_AllotmentGardens_r500.tif`

layername: `egv_404`

English name: Fractional cover of Allotment gardens within the 0.5 km landscape

Latvian name: Vasarnīcu kompleksu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.405 General_AllotmentGardens_r1250

filename: `General_AllotmentGardens_r1250.tif`

layername: `egv_405`

English name: Fractional cover of Allotment gardens within the 1.25 km landscape

Latvian name: Vasarnīcu kompleksu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.406 General_AllotmentGardens_r3000

filename: `General_AllotmentGardens_r3000.tif`

layername: `egv_406`

English name: Fractional cover of Allotment gardens within the 3 km landscape

Latvian name: Vasarnīcu kompleksu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.407 General_AllotmentGardens_r10000

filename: `General_AllotmentGardens_r10000.tif`

layername: `egv_407`

English name: Fractional cover of Allotment gardens within the 10 km landscape

Latvian name: Vasarnīcu kompleksu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.408 General_BareSoilQuarry_cell

filename: `General_BareSoilQuarry_cell.tif`

layername: `egv_408`

English name: Fractional cover of areas with Bare Soil, Quarries within the analysis cell (1 ha)

Latvian name: Atklātas augsnes un karjeru platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.409 General_BareSoilQuarry_r500

filename: `General_BareSoilQuarry_r500.tif`

layername: `egv_409`

English name: Fractional cover of areas with Bare Soil, Quarries within the 0.5 km landscape

Latvian name: Atklātas augsnes un karjeru platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.410 General_BareSoilQuarry_r1250

filename: `General_BareSoilQuarry_r1250.tif`

layername: `egv_410`

English name: Fractional cover of areas with Bare Soil, Quarries within the 1.25 km landscape

Latvian name: Atklātas augsnes un karjeru platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.411 General_BareSoilQuarry_r3000

filename: `General_BareSoilQuarry_r3000.tif`

layername: `egv_411`

English name: Fractional cover of areas with Bare Soil, Quarries within the 3 km landscape

Latvian name: Atklātas augsnes un karjeru platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.412 General_BareSoilQuarry_r10000

filename: `General_BareSoilQuarry_r10000.tif`

layername: `egv_412`

English name: Fractional cover of areas with Bare Soil, Quarries within the 10 km landscape

Latvian name: Atklātas augsnes un karjeru platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.413 General_Builtup_cell

filename: `General_Builtup_cell.tif`

layername: `egv_413`

English name: Fractional cover of Built-Up areas within the analysis cell (1 ha)

Latvian name: Apbūves platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.414 General_Builtup_r500

filename: `General_Builtup_r500.tif`

layername: `egv_414`

English name: Fractional cover of Built-Up areas within the 0.5 km landscape

Latvian name: Apbūves platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.415 General_Builtup_r1250

filename: `General_Builtup_r1250.tif`

layername: `egv_415`

English name: Fractional cover of Built-Up areas within the 1.25 km landscape

Latvian name: Apbūves platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.416 General_Builtup_r3000

filename: `General_Builtup_r3000.tif`

layername: `egv_416`

English name: Fractional cover of Built-Up areas within the 3 km landscape

Latvian name: Apbūves platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.417 General_Builtup_r10000

filename: `General_Builtup_r10000.tif`

layername: `egv_417`

English name: Fractional cover of Built-Up areas within the 10 km landscape

Latvian name: Apbūves platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.418 General_Farmland_cell

filename: `General_Farmland_cell.tif`

layername: `egv_418`

English name: Fractional cover of Farmland within the analysis cell (1 ha)

Latvian name: Lauksaimniecībā izmantojamo zemju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.419 General_Farmland_r500

filename: `General_Farmland_r500.tif`

layername: `egv_419`

English name: Fractional cover of Farmland within the 0.5 km landscape

Latvian name: Lauksaimniecībā izmantojamo zemju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.420 General_Farmland_r1250

filename: `General_Farmland_r1250.tif`

layername: `egv_420`

English name: Fractional cover of Farmland within the 1.25 km landscape

Latvian name: Lauksaimniecībā izmantojamo zemju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.421 General_Farmland_r3000

filename: `General_Farmland_r3000.tif`

layername: `egv_421`

English name: Fractional cover of Farmland within the 3 km landscape

Latvian name: Lauksaimniecībā izmantojamo zemju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.422 General_Farmland_r10000

filename: `General_Farmland_r10000.tif`

layername: `egv_422`

English name: Fractional cover of Farmland within the 10 km landscape

Latvian name: Lauksaimniecībā izmantojamo zemju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.423 General_ForestsWithoutInventory_cell

filename: `General_ForestsWithoutInventory_cell.tif`

layername: `egv_423`

English name: Fractional cover of Forests Without Inventory within the analysis cell (1 ha)

Latvian name: Netaksēto mežu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.424 General_ForestsWithoutInventory_r500

filename: `General_ForestsWithoutInventory_r500.tif`

layername: `egv_424`

English name: Fractional cover of Forests Without Inventory within the 0.5 km landscape

Latvian name: Netaksēto mežu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.425 General_ForestsWithoutInventory_r1250

filename: `General_ForestsWithoutInventory_r1250.tif`

layername: `egv_425`

English name: Fractional cover of Forests Without Inventory within the 1.25 km landscape

Latvian name: Netaksēto mežu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.426 General_ForestsWithoutInventory_r3000

filename: `General_ForestsWithoutInventory_r3000.tif`

layername: `egv_426`

English name: Fractional cover of Forests Without Inventory within the 3 km landscape

Latvian name: Netaksēto mežu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.427 General_ForestsWithoutInventory_r10000

filename: `General_ForestsWithoutInventory_r10000.tif`

layername: `egv_427`

English name: Fractional cover of Forests Without Inventory within the 10 km landscape

Latvian name: Netaksēto mežu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.428 General_GardensOrchards_cell

filename: `General_GardensOrchards_cell.tif`

layername: `egv_428`

English name: Fractional cover of Allotment gardens, Orchards within the analysis cell (1 ha)

Latvian name: Vasarnīcu kompleksu un augļudārzu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.429 General_GardensOrchards_r500

filename: `General_GardensOrchards_r500.tif`

layername: `egv_429`

English name: Fractional cover of Allotment gardens, Orchards within the 0.5 km landscape

Latvian name: Vasarnīcu kompleksu un augļudārzu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.430 General_GardensOrchards_r1250

filename: `General_GardensOrchards_r1250.tif`

layername: `egv_430`

English name: Fractional cover of Allotment gardens, Orchards within the 1.25 km landscape

Latvian name: Vasarnīcu kompleksu un augļudārzu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.431 General_GardensOrchards_r3000

filename: `General_GardensOrchards_r3000.tif`

layername: `egv_431`

English name: Fractional cover of Allotment gardens, Orchards within the 3 km landscape

Latvian name: Vasarnīcu kompleksu un augļudārzu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.432 General_GardensOrchards_r10000

filename: `General_GardensOrchards_r10000.tif`

layername: `egv_432`

English name: Fractional cover of Allotment gardens, Orchards within the 10 km landscape

Latvian name: Vasarnīcu kompleksu un augļudārzu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.433 General_Roads_cell

filename: `General_Roads_cell.tif`

layername: `egv_433`

English name: Fractional cover of Roads within the analysis cell (1 ha)

Latvian name: Ceļu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.434 General_ShrubsOrchards_cell

filename: `General_ShrubsOrchards_cell.tif`

layername: `egv_434`

English name: Fractional cover of Shrubs, Young stands, Orchards within the analysis cell (1 ha)

Latvian name: Krūmāju, jaunaudžu un augļudārzu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.435 General_ShrubsOrchards_r500

filename: `General_ShrubsOrchards_r500.tif`

layername: `egv_435`

English name: Fractional cover of Shrubs, Young stands, Orchards within the 0.5 km landscape

Latvian name: Krūmāju, jaunaudžu un augļudārzu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.436 General_ShrubsOrchards_r1250

filename: `General_ShrubsOrchards_r1250.tif`

layername: `egv_436`

English name: Fractional cover of Shrubs, Young stands, Orchards within the 1.25 km landscape

Latvian name: Krūmāju, jaunaudžu un augļudārzu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.437 General_ShrubsOrchards_r3000

filename: `General_ShrubsOrchards_r3000.tif`

layername: `egv_437`

English name: Fractional cover of Shrubs, Young stands, Orchards within the 3 km landscape

Latvian name: Krūmāju, jaunaudžu un augļudārzu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.438 General_ShrubsOrchards_r10000

filename: `General_ShrubsOrchards_r10000.tif`

layername: `egv_438`

English name: Fractional cover of Shrubs, Young stands, Orchards within the 10 km landscape

Latvian name: Krūmāju, jaunaudžu un augļudārzu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.439 General_ShrubsOrchardsGardens_cell

filename: `General_ShrubsOrchardsGardens_cell.tif`

layername: `egv_439`

English name: Fractional cover of Shrubs, Young stands, Orchards, Allotment gardens within the analysis cell (1 ha)

Latvian name: Krūmāju, jaunaudžu, augļudārzu un vasarnīcu kompleksu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.440 General_ShrubsOrchardsGardens_r500

filename: `General_ShrubsOrchardsGardens_r500.tif`

layername: `egv_440`

English name: Fractional cover of Shrubs, Young stands, Orchards, Allotment gardens within the 0.5 km landscape

Latvian name: Krūmāju, jaunaudžu, augļudārzu un vasarnīcu kompleksu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.441 General_ShrubsOrchardsGardens_r1250

filename: `General_ShrubsOrchardsGardens_r1250.tif`

layername: `egv_441`

English name: Fractional cover of Shrubs, Young stands, Orchards, Allotment gardens within the 1.25 km landscape

Latvian name: Krūmāju, jaunaudžu, augļudārzu un vasarnīcu kompleksu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.442 General_ShrubsOrchardsGardens_r3000

filename: `General_ShrubsOrchardsGardens_r3000.tif`

layername: `egv_442`

English name: Fractional cover of Shrubs, Young stands, Orchards, Allotment gardens within the 3 km landscape

Latvian name: Krūmāju, jaunaudžu, augļudārzu un vasarnīcu kompleksu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.443 General_ShrubsOrchardsGardens_r10000

filename: `General_ShrubsOrchardsGardens_r10000.tif`

layername: `egv_443`

English name: Fractional cover of Shrubs, Young stands, Orchards, Allotment gardens within the 10 km landscape

Latvian name: Krūmāju, jaunaudžu, augļudārzu un vasarnīcu kompleksu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.444 General_SwampsMiresBogsHelophytes_cell

filename: `General_SwampsMiresBogsHelophytes_cell.tif`

layername: `egv_444`

English name: Fractional cover of Swamps, Mires, Bogs, Reed-, Sedge-, Rush- Beds within the analysis cell (1 ha)

Latvian name: Purvu, niedrāju, grīslāju, meldrāju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.445 General_SwampsMiresBogsHelophytes_r500

filename: `General_SwampsMiresBogsHelophytes_r500.tif`

layername: `egv_445`

English name: Fractional cover of Swamps, Mires, Bogs, Reed-, Sedge-, Rush- Beds within the 0.5 km landscape

Latvian name: Purvu, niedrāju, grīslāju, meldrāju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.446 General_SwampsMiresBogsHelophytes_r1250

filename: `General_SwampsMiresBogsHelophytes_r1250.tif`

layername: `egv_446`

English name: Fractional cover of Swamps, Mires, Bogs, Reed-, Sedge-, Rush- Beds within the 1.25 km landscape

Latvian name: Purvu, niedrāju, grīslāju, meldrāju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.447 General_SwampsMiresBogsHelophytes_r3000

filename: `General_SwampsMiresBogsHelophytes_r3000.tif`

layername: `egv_447`

English name: Fractional cover of Swamps, Mires, Bogs, Reed-, Sedge-, Rush- Beds within the 3 km landscape

Latvian name: Purvu, niedrāju, grīslāju, meldrāju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.448 General_SwampsMiresBogsHelophytes_r10000

filename: `General_SwampsMiresBogsHelophytes_r10000.tif`

layername: `egv_448`

English name: Fractional cover of Swamps, Mires, Bogs, Reed-, Sedge-, Rush- Beds within the 10 km landscape

Latvian name: Purvu, niedrāju, grīslāju, meldrāju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.449 General_Trees_cell

filename: `General_Trees_cell.tif`

layername: `egv_449`

English name: Fractional cover of Trees, Shrubs, Clear-cuts within the analysis cell (1 ha)

Latvian name: Koku, krūmu un izcirtumu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.450 General_Trees_r500

filename: `General_Trees_r500.tif`

layername: `egv_450`

English name: Fractional cover of Trees, Shrubs, Clear-cuts within the 0.5 km landscape

Latvian name: Koku, krūmu un izcirtumu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.451 General_Trees_r1250

filename: `General_Trees_r1250.tif`

layername: `egv_451`

English name: Fractional cover of Trees, Shrubs, Clear-cuts within the 1.25 km landscape

Latvian name: Koku, krūmu un izcirtumu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.452 General_Trees_r3000

filename: `General_Trees_r3000.tif`

layername: `egv_452`

English name: Fractional cover of Trees, Shrubs, Clear-cuts within the 3 km landscape

Latvian name: Koku, krūmu un izcirtumu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.453 General_Trees_r10000

filename: `General_Trees_r10000.tif`

layername: `egv_453`

English name: Fractional cover of Trees, Shrubs, Clear-cuts within the 10 km landscape

Latvian name: Koku, krūmu un izcirtumu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.454 General_TreesOutsideForests_cell

filename: `General_TreesOutsideForests_cell.tif`

layername: `egv_454`

English name: Fractional cover of Tree covered areas Outside Forests within the analysis cell (1 ha)

Latvian name: Ar kokiem klāto teritoriju ārpus mežiem platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.455 General_TreesOutsideForests_r500

filename: `General_TreesOutsideForests_r500.tif`

layername: `egv_455`

English name: Fractional cover of Tree covered areas Outside Forests within the 0.5 km landscape

Latvian name: Ar kokiem klāto teritoriju ārpus mežiem platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.456 General_TreesOutsideForests_r1250

filename: `General_TreesOutsideForests_r1250.tif`

layername: `egv_456`

English name: Fractional cover of Tree covered areas Outside Forests within the 1.25 km landscape

Latvian name: Ar kokiem klāto teritoriju ārpus mežiem platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.457 General_TreesOutsideForests_r3000

filename: `General_TreesOutsideForests_r3000.tif`

layername: `egv_457`

English name: Fractional cover of Tree covered areas Outside Forests within the 3 km landscape

Latvian name: Ar kokiem klāto teritoriju ārpus mežiem platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.458 General_TreesOutsideForests_r10000

filename: `General_TreesOutsideForests_r10000.tif`

layername: `egv_458`

English name: Fractional cover of Tree covered areas Outside Forests within the 10 km landscape

Latvian name: Ar kokiem klāto teritoriju ārpus mežiem platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.459 General_Water_cell

filename: `General_Water_cell.tif`

layername: `egv_459`

English name: Fractional cover of Waterbodies within the analysis cell (1 ha)

Latvian name: Ūdenstilpju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.460 General_Water_r500

filename: `General_Water_r500.tif`

layername: `egv_460`

English name: Fractional cover of Waterbodies within the 0.5 km landscape

Latvian name: Ūdenstilpju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.461 General_Water_r1250

filename: `General_Water_r1250.tif`

layername: `egv_461`

English name: Fractional cover of Waterbodies within the 1.25 km landscape

Latvian name: Ūdenstilpju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.462 General_Water_r3000

filename: `General_Water_r3000.tif`

layername: `egv_462`

English name: Fractional cover of Waterbodies within the 3 km landscape

Latvian name: Ūdenstilpju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.463 General_Water_r10000

filename: `General_Water_r10000.tif`

layername: `egv_463`

English name: Fractional cover of Waterbodies within the 10 km landscape

Latvian name: Ūdenstilpju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.464 Wetlands_Bogs_cell

filename: `Wetlands_Bogs_cell.tif`

layername: `egv_464`

English name: Fractional cover of Raised Bogs within the analysis cell (1 ha)

Latvian name: Augsto purvu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.465 Wetlands_Bogs_r500

filename: `Wetlands_Bogs_r500.tif`

layername: `egv_465`

English name: Fractional cover of Raised Bogs within the 0.5 km landscape

Latvian name: Augsto purvu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.466 Wetlands_Bogs_r1250

filename: `Wetlands_Bogs_r1250.tif`

layername: `egv_466`

English name: Fractional cover of Raised Bogs within the 1.25 km landscape

Latvian name: Augsto purvu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.467 Wetlands_Bogs_r3000

filename: `Wetlands_Bogs_r3000.tif`

layername: `egv_467`

English name: Fractional cover of Raised Bogs within the 3 km landscape

Latvian name: Augsto purvu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.468 Wetlands_Bogs_r10000

filename: `Wetlands_Bogs_r10000.tif`

layername: `egv_468`

English name: Fractional cover of Raised Bogs within the 10 km landscape

Latvian name: Augsto purvu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.469 Wetlands_Mires_cell

filename: `Wetlands_Mires_cell.tif`

layername: `egv_469`

English name: Fractional cover of Transitional Mires within the analysis cell (1 ha)

Latvian name: Pārejas purvu platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.470 Wetlands_Mires_r500

filename: `Wetlands_Mires_r500.tif`

layername: `egv_470`

English name: Fractional cover of Transitional Mires within the 0.5 km landscape

Latvian name: Pārejas purvu platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.471 Wetlands_Mires_r1250

filename: `Wetlands_Mires_r1250.tif`

layername: `egv_471`

English name: Fractional cover of Transitional Mires within the 1.25 km landscape

Latvian name: Pārejas purvu platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.472 Wetlands_Mires_r3000

filename: `Wetlands_Mires_r3000.tif`

layername: `egv_472`

English name: Fractional cover of Transitional Mires within the 3 km landscape

Latvian name: Pārejas purvu platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.473 Wetlands_Mires_r10000

filename: `Wetlands_Mires_r10000.tif`

layername: `egv_473`

English name: Fractional cover of Transitional Mires within the 10 km landscape

Latvian name: Pārejas purvu platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.474 Wetlands_ReedSedgeRushBeds_cell

filename: `Wetlands_ReedSedgeRushBeds_cell.tif`

layername: `egv_474`

English name: Fractional cover of Reed-, Sedge-, Rush-, Beds within the analysis cell (1 ha)

Latvian name: Niedrāju, grīslāju, meldrāju platības īpatsvars analīzes šūnā (1 ha)

Procedure:

```
# libs ----
```

6.475 Wetlands_ReedSedgeRushBeds_r500

filename: `Wetlands_ReedSedgeRushBeds_r500.tif`

layername: `egv_475`

English name: Fractional cover of Reed-, Sedge-, Rush-, Beds within the 0.5 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju platības īpatsvars 0,5 km ainavā

Procedure:

```
# libs ----
```

6.476 Wetlands_ReedSedgeRushBeds_r1250

filename: `Wetlands_ReedSedgeRushBeds_r1250.tif`

layername: `egv_476`

English name: Fractional cover of Reed-, Sedge-, Rush-, Beds within the 1.25 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju platības īpatsvars 1,25 km ainavā

Procedure:

```
# libs ----
```

6.477 Wetlands_ReedSedgeRushBeds_r3000

filename: `Wetlands_ReedSedgeRushBeds_r3000.tif`

layername: `egv_477`

English name: Fractional cover of Reed-, Sedge-, Rush-, Beds within the 3 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju platības īpatsvars 3 km ainavā

Procedure:

```
# libs ----
```

6.478 Wetlands_ReedSedgeRushBeds_r10000

filename: `Wetlands_ReedSedgeRushBeds_r10000.tif`

layername: `egv_478`

English name: Fractional cover of Reed-, Sedge-, Rush-, Beds within the 10 km landscape

Latvian name: Niedrāju, grīslāju, meldrāju platības īpatsvars 10 km ainavā

Procedure:

```
# libs ----
```

6.479 EO_NDMI-LYmed-average_cell

filename: `EO_NDMI-LYmed-average_cell.tif`

layername: `egv_479`

English name: Median vegetation water content (NDMI) for the last year within the analysis cell (1 ha)

Latvian name: Mediānā pēdējā gada ūdens satura veģetācijā indeksa (NDMI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Last year is 2024.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDMI-LYmed-average_cell.tif ----
egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDMI-LYmed-average_cell.tif",
                 layername = "egv_479",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.480 EO_NDMI-LYmedian-iqr_cell

filename: `EO_NDMI-LYmedian-iqr_cell.tif`

layername: `egv_480`

English name: Spatial variability of last year's median vegetation water content (NDMI) within the analysis cell (1 ha)

Latvian name: Telpiskā variabilitāte pēdējā gada mediānajai ūdens saturam veģetācijā indeksa (NDMI) vērtībai, starpkvartiļu apgabals analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. First Q1 and then Q3 is calculated for every cell with `egvtools::input2egv()`. Finally, subtracting Q1 from Q3 and writing final raster with specified layername. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Last year is 2024.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDMI-LYmedian-iqr_cell.tif ----

p25rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q1",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p25.tif",
                 layername = "egv_480",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p25rez_r=rast("./RasterGrids_100m/2024/draza_p25.tif")


p75rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q3",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p75.tif",
                 layername = "egv_480",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p75rez_r=rast("./RasterGrids_100m/2024/draza_p75.tif")
```

```
iqr_rez=p75rez_r-p25rez_r
iqr_rez
plot(iqr_rez)

writeRaster(iqr_rez,
            "./RasterGrids_100m/2024/RAW/EO_NDMI-LYmedian-iqr_cell.tif",
            overwrite=TRUE)

unlink("./RasterGrids_100m/2024/draza_p75.tif")
unlink("./RasterGrids_100m/2024/draza_p25.tif")
```

6.481 EO_NDMI-STiqr-median_cell

filename: `EO_NDMI-STiqr-median_cell.tif`

layername: `egv_481`

English name: Average short-term seasonality of vegetation water content (NDMI) within the analysis cell (1 ha)

Latvian name: Sezonalitāte pēdējo piecu gadu vidējam ūdens satura veģetācijā indeksa (NDMI) vērtībai, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# EO_NDMI-STiqr-median_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-STiqr.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDMI-STiqr-median_cell.tif",
                 layername = "egv_481",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.482 EO_NDMI-STmedian-average_cell

filename: `EO_NDMI-STmedian-average_cell.tif`

layername: `egv_482`

English name: Median short-term vegetation water content (NDMI) within the analysis cell (1 ha)

Latvian name: Mediānā pēdējo piecu gadu ūdens satura veģetācijā indeksa (NDMI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDMI-STmedian-average_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDMI-STmedian-average_cell.tif",
                 layername = "egv_482",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.483 EO_NDMI-STmedian-iqr_cell

filename: `EO_NDMI-STmedian-iqr_cell.tif`

layername: `egv_483`

English name: Spatial variability of short-term median vegetation water content (NDMI) within the analysis cell (1 ha)

Latvian name: Telpiskā variabilitāte pēdējo piecu gadu mediānajai ūdens saturam veģetācijā indeksa (NDMI) vērtībai, starpkvartiļu apgabals analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. First Q1 and then Q3 is calculated for every cell with `egvtools::input2egv()`. Finally, subtracting Q1 from Q3 and writing final raster with specified layername. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term corresponds to last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# EO_NDMI-STmedian-iqr_cell.tif ----


p25rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q1",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p25.tif",
                 layername = "egv_483",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p25rez_r=rast("./RasterGrids_100m/2024/draza_p25.tif")


p75rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q3",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p75.tif",
                 layername = "egv_483",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p75rez_r=rast("./RasterGrids_100m/2024/draza_p75.tif")

iqr_rez=p75rez_r-p25rez_r
iqr_rez
plot(iqr_rez)

writeRaster(iqr_rez,
            "./RasterGrids_100m/2024/RAW/EO_NDMI-STmedian-iqr_cell.tif",
            overwrite=TRUE)

unlink("./RasterGrids_100m/2024/draza_p75.tif")
```

```
unlink("./RasterGrids_100m/2024/draza_p25.tif")
```

6.484 EO_NDMI-STp25-min_cell

filename: `EO_NDMI-STp25-min_cell.tif`

layername: `egv_484`

English name: Minimum short-term 25th percentile of vegetation water content (NDMI) within the analysis cell (1 ha)

Latvian name: Minimālā 25. procentiles pēdējo piecu gadu ūdens satura veģetācijā indeksa (NDMI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Minimum value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# EO_NDMI-STp25-min_cell.tif ----


egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-STp25.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "min",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDMI-STp25-min_cell.tif",
                 layername = "egv_484",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.485 EO_NDMI-STp75-max_cell

filename: `EO_NDMI-STp75-max_cell.tif`

layername: `egv_485`

English name: Maximum short-term 75th percentile of vegetation water content (NDMI) within the analysis cell (1 ha)

Latvian name: Maksimālā 75. procentiles pēdējo piecu gadu ūdens satura veģetācijā indeksa (NDMI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Maximum value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDMI-STp75-max_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDMI-STp75.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "min",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDMI-STp75-max_cell.tif",
                 layername = "egv_485",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.486 EO_NDVI-LYmedian-average_cell

filename: `EO_NDVI-LYmedian-average_cell.tif`

layername: `egv_486`

English name: Median vegetation index (NDVI) for the last year within the analysis cell (1 ha)

Latvian name: Mediānā pēdējā gada veģetācijas indeksa (NDVI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Last year is 2024.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# EO_NDVI-LYmedian-average_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDVI-LYmedian-average_cell.tif",
                 layername = "egv_486",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.487 EO_NDVI-LYmedian-iqr_cell

filename: `EO_NDVI-LYmedian-iqr_cell.tif`

layername: `egv_487`

English name: Spatial variability of last year's median vegetation index (NDVI) within the analysis cell (1 ha)

Latvian name: Telpiskā variabilitāte pēdējā gada mediānajai veģetācijas indeksa (NDVI) vērtībai, starpkvartiļu apgabals analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. First Q1 and then Q3 is calculated for every cell with `egvtools::input2egv()`. Finally, subtracting Q1 from Q3 and writing final raster with specified layername. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Last year is 2024.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# EO_NDVI-LYmedian-iqr_cell.tif ----


p25rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q1",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p25.tif",
```

```
                 layername = "egv_487",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p25rez_r=rast("./RasterGrids_100m/2024/draza_p25.tif")


p75rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q3",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p75.tif",
                 layername = "egv_487",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p75rez_r=rast("./RasterGrids_100m/2024/draza_p75.tif")

iqr_rez=p75rez_r-p25rez_r
iqr_rez
plot(iqr_rez)

writeRaster(iqr_rez,
            "./RasterGrids_100m/2024/RAW/EO_NDVI-LYmedian-iqr_cell.tif",
            overwrite=TRUE)

unlink("./RasterGrids_100m/2024/draza_p75.tif")
unlink("./RasterGrids_100m/2024/draza_p25.tif")
```

6.488 EO_NDVI-STiqr-median_cell

filename: `EO_NDVI-STiqr-median_cell.tif`

layername: `egv_488`

English name: Average short-term seasonality of vegetation index (NDVI) within the analysis cell (1 ha)

Latvian name: Sezonalitāte pēdējo piecu gadu vidējam veģetācijas indeksa (NDVI) vērtībai, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDVI-STiqr-median_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-STiqr.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDVI-STiqr-median_cell.tif",
                 layername = "egv_488",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.489 EO_NDVI-STmedian-average_cell

filename: `EO_NDVI-STmedian-average_cell.tif`

layername: `egv_489`

English name: Median short-term vegetation index (NDVI) within the analysis cell (1 ha)

Latvian name: Mediānā pēdējo piecu gadu veģetācijas indeksa (NDVI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDVI-STmedian-average_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
```

```
                 outfilename = "EO_NDVI-STmedian-average_cell.tif",
                 layername = "egv_489",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.490 EO_NDVI-STmedian-iqr_cell

filename: `EO_NDVI-STmedian-iqr_cell.tif`

layername: `egv_490`

English name: Spatial variability of short-term median vegetation index (NDVI) within the analysis cell (1 ha)

Latvian name: Telpiskā variabilitāte pēdējo piecu gadu mediānajai veģetācijas indeksa (NDVI) vērtībai, starpkvartiļu apgabals analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. First Q1 and then Q3 is calculated for every cell with `egvtools::input2egv()`. Finally, subtracting Q1 from Q3 and writing final raster with specified layername. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term corresponds to last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# EO_NDVI-STmedian-iqr_cell.tif ----


p25rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q1",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p25.tif",
                 layername = "egv_490",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p25rez_r=rast("./RasterGrids_100m/2024/draza_p25.tif")
```

```
p75rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q3",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p75.tif",
                 layername = "egv_490",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p75rez_r=rast("./RasterGrids_100m/2024/draza_p75.tif")

iqr_rez=p75rez_r-p25rez_r
iqr_rez
plot(iqr_rez)

writeRaster(iqr_rez,
            "./RasterGrids_100m/2024/RAW/EO_NDVI-STmedian-iqr_cell.tif",
            overwrite=TRUE)

unlink("./RasterGrids_100m/2024/draza_p75.tif")
unlink("./RasterGrids_100m/2024/draza_p25.tif")
```

6.491 EO_NDVI-STp25-min_cell

filename: `EO_NDVI-STp25-min_cell.tif`

layername: `egv_491`

English name: Minimum short-term 25th percentile of vegetation index (NDVI) within the analysis cell (1 ha)

Latvian name: Minimālā 25. procentiles pēdējo piecu gadu veģetācijas indeksa (NDVI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Minimum value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDVI-STp25-min_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-STp25.tif",
```

```
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "min",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDVI-STp25-min_cell.tif",
                 layername = "egv_491",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.492 EO_NDVI-STp75-max_cell

filename: `EO_NDVI-STp75-max_cell.tif`

layername: `egv_492`

English name: Maximum short-term 75th percentile of vegetation index (NDVI) within the analysis cell (1 ha)

Latvian name: Maksimālā 75. procentiles pēdējo piecu gadu veģetācijas indeksa (NDVI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Maximum value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDVI-STp75-max_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDVI-STp75.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "min",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDVI-STp75-max_cell.tif",
                 layername = "egv_492",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.493 EO_NDWI-LYmedian-average_cell

filename: `EO_NDWI-LYmedian-average_cell.tif`

layername: `egv_493`

English name: Median water index (NDWI) for the last year within the analysis cell (1 ha)

Latvian name: Mediānā pēdējā gada ūdens indeksa (NDWI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Last year is 2024.

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDWI-LYmedian-average_cell.tif",
                 layername = "egv_493",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.494 EO_NDWI-LYmedian-iqr_cell

filename: `EO_NDWI-LYmedian-iqr_cell.tif`

layername: `egv_494`

English name: Spatial variability of last year's median water index (NDWI) within the analysis cell (1 ha)

Latvian name: Telpiskā variabilitāte pēdējā gada mediānajai ūdens indeksa (NDWI) vērtībai, starpkvartiļu apgabals analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. First Q1 and then Q3 is calculated for every cell with `egvtools::input2egv()`. Finally, subtracting Q1 from Q3 and writing final raster with specified layername. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Last year is 2024.

```r
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDWI-LYmedian-iqr_cell.tif ----


p25rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q1",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p25.tif",
                 layername = "egv_494",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p25rez_r=rast("./RasterGrids_100m/2024/draza_p25.tif")


p75rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-LYmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q3",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p75.tif",
                 layername = "egv_494",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p75rez_r=rast("./RasterGrids_100m/2024/draza_p75.tif")

iqr_rez=p75rez_r-p25rez_r
iqr_rez
plot(iqr_rez)

writeRaster(iqr_rez,
            "./RasterGrids_100m/2024/RAW/EO_NDWI-LYmedian-iqr_cell.tif",
            overwrite=TRUE)

unlink("./RasterGrids_100m/2024/draza_p75.tif")
unlink("./RasterGrids_100m/2024/draza_p25.tif")
```

6.495 EO_NDWI-STiqr-median_cell

filename: `EO_NDWI-STiqr-median_cell.tif`

layername: `egv_495`

English name: Average short-term seasonality of water index (NDWI) within the analysis cell (1 ha)

Latvian name: Sezonalitāte pēdējo piecu gadu vidējam ūdens indeksa (NDWI) vērtībai, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDWI-STiqr-median_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-STiqr.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDWI-STiqr-median_cell.tif",
                 layername = "egv_495",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.496 EO_NDWI-STmedian-average_cell

filename: `EO_NDWI-STmedian-average_cell.tif`

layername: `egv_496`

English name: Median short-term water index (NDWI) within the analysis cell (1 ha)

Latvian name: Mediānā pēdējo piecu gadu ūdens indeksa (NDWI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Arithmetic mean value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDWI-STmedian-average_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "average",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDWI-STmedian-average_cell.tif",
                 layername = "egv_496",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.497 EO_NDWI-STmedian-iqr_cell

filename: `EO_NDWI-STmedian-iqr_cell.tif`

layername: `egv_497`

English name: Spatial variability of short-term median water index (NDWI) within the analysis cell (1 ha)

Latvian name: Telpiskā variabilitāte pēdējo piecu gadu mediānajai ūdens indeksa (NDWI) vērtībai, starpkvartiļu apgabals analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. First Q1 and then Q3 is calculated for every cell with `egvtools::input2egv()`. Finally, subtracting Q1 from Q3 and writing final raster with specified layername. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term corresponds to last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# EO_NDWI-STmedian-iqr_cell.tif ----
```

```
p25rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q1",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p25.tif",
                 layername = "egv_497",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p25rez_r=rast("./RasterGrids_100m/2024/draza_p25.tif")


p75rez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-STmedian.tif
    ↪ ",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "q3",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/",
                 outfilename = "draza_p75.tif",
                 layername = "egv_497",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
p75rez_r=rast("./RasterGrids_100m/2024/draza_p75.tif")

iqr_rez=p75rez_r-p25rez_r
iqr_rez
plot(iqr_rez)

writeRaster(iqr_rez,
            "./RasterGrids_100m/2024/RAW/EO_NDWI-STmedian-iqr_cell.tif",
            overwrite=TRUE)

unlink("./RasterGrids_100m/2024/draza_p75.tif")
unlink("./RasterGrids_100m/2024/draza_p25.tif")
```

6.498 EO_NDWI-STp25-min_cell

filename: `EO_NDWI-STp25-min_cell.tif`

layername: `egv_498`

English name: Minimum short-term 25th percentile of water index (NDWI) within the analysis cell (1 ha)

Latvian name: Minimālā 25. procentiles pēdējo piecu gadu ūdens indeksa (NDWI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Minimum value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EO_NDWI-STp25-min_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-STp25.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "min",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDWI-STp25-min_cell.tif",
                 layername = "egv_498",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.499 EO_NDWI-STp75-max_cell

filename: `EO_NDWI-STp75-max_cell.tif`

layername: `egv_499`

English name: Maximum short-term 75th percentile of water index (NDWI) within the analysis cell (1 ha)

Latvian name: Maksimālā 75. procentiles pēdējo piecu gadu ūdens indeksa (NDWI) vērtība, vidējais analīzes šūnā (1 ha)

Procedure: Directly follows preprocessing. Maximum value at analysis cell calculated with `egvtools::input2egv()`. To protect against possible data loss at edge cells, inverse distance weighted (power = 2) gap filling is implemented. Short-term is last five years (2020-2024).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# EO_NDWI-STp75-max_cell.tif ----

egvrez=input2egv(input="./Geodata/2024/S2indices/Mosaics/EO_NDWI-STp75.tif",
                 egv_template= "./Templates/TemplateRasters/LV100m_10km.tif",
                 summary_function = "min",
                 missing_job = "FillOutput",
                 outlocation = "./RasterGrids_100m/2024/RAW/",
                 outfilename = "EO_NDWI-STp75-max_cell.tif",
                 layername = "egv_499",
                 idw_weight = 2,
                 plot_gaps = FALSE,
                 plot_final = FALSE)
egvrez
```

6.500 SoilChemistry_ESDAC-CN_cell

filename: `SoilChemistry_ESDAC-CN_cell.tif`

layername: `egv_500`

English name: Average value of Topsoil Carbon-Nitrogen ratio (ESDAC v2.0) within the analysis cell (1 ha)

Latvian name: Augsnes virskārtas oglekļa-slāpekļa attiecība (ESDAC v2.0) analīzes šūnā (1 ha)

Procedure: Directly derived from Soil chemistry. Processed with `egvtools::downscale2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border and `smooth = FALSE` to keep as original values as reasonable (there is bilinear interpolation involved when projecting from 500 m to 100 m resolution of different CRS).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    (egvtools)}


# CN ----

egv=downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = "./Geodata/2024/Soils/ESDAC/chemistry/chemistry/CN/CN.tif",
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = "SoilChemistry_ESDAC-CN_cell.tif",
  layer_name    = "egv_500",
  fill_gaps     = TRUE,
  smooth        = FALSE,
```

```
  plot_result   = TRUE)
egv
```

6.501 SoilChemistry_ESDAC-CaCo3_cell

filename: `SoilChemistry_ESDAC-CaCo3_cell.tif`

layername: `egv_501`

English name: Average value of Topsoil Calcium Carbonates Content (ESDAC v2.0) within the analysis cell (1 ha)

Latvian name: Augsnes virskārtas kalcija karbonātu apjoms (ESDAC v2.0) analīzes šūnā (1 ha)

Procedure: Directly derived from Soil chemistry. Processed with `egvtools::downscale2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border and `smooth = FALSE` to keep as original values as reasonable (there is bilinear interpolation involved when projecting from 500 m to 100 m resolution of different CRS).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# CaCO3 ----


egv=downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = "./Geodata/2024/Soils/ESDAC/chemistry/chemistry/Caco3/CaCO3
    ↪ .tif",
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = "SoilChemistry_ESDAC-CaCo3_cell.tif",
  layer_name    = "egv_501",
  fill_gaps     = TRUE,
  smooth        = FALSE,
  plot_result   = TRUE)
egv
```

6.502 SoilChemistry_ESDAC-K_cell

filename: `SoilChemistry_ESDAC-K_cell.tif`

layername: `egv_502`

English name: Average value of Topsoil Sodium Content (ESDAC v2.0) within the analysis cell (1 ha)

Latvian name: Augsnes virskārtas kālija apjoms (ESDAC v2.0) analīzes šūnā (1 ha)

Procedure: Directly derived from Soil chemistry. Processed with `egvtools::`
↪ `downscale2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border and `smooth = FALSE` to keep as original values as reasonable (there is bilinear interpolation involved when projecting from 500 m to 100 m resolution of different CRS).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# K ----

egv=downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = "./Geodata/2024/Soils/ESDAC/chemistry/chemistry/K/K.tif",
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = "SoilChemistry_ESDAC-K_cell.tif",
  layer_name    = "egv_502",
  fill_gaps     = TRUE,
  smooth        = FALSE,
  plot_result   = TRUE)
egv
```

6.503 SoilChemistry_ESDAC-N_cell

filename: `SoilChemistry_ESDAC-N_cell.tif`

layername: `egv_503`

English name: Average value of Topsoil Nitrogen Content (ESDAC v2.0) within the analysis cell (1 ha)

Latvian name: Augsnes virskārtas slāpekļa apjoms (ESDAC v2.0) analīzes šūnā (1 ha)

Procedure: Directly derived from Soil chemistry. Processed with `egvtools::`
↪ `downscale2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border and `smooth = FALSE` to keep as original values as reasonable (there is bilinear interpolation involved when projecting from 500 m to 100 m resolution of different CRS).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# N ----

egv=downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = "./Geodata/2024/Soils/ESDAC/chemistry/chemistry/N/N.tif",
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = "SoilChemistry_ESDAC-N_cell.tif",
  layer_name    = "egv_503",
  fill_gaps     = TRUE,
  smooth        = FALSE,
  plot_result   = TRUE)
egv
```

6.504 SoilChemistry_ESDAC-P_cell

filename: `SoilChemistry_ESDAC-P_cell.tif`

layername: `egv_504`

English name: Average value of Topsoil Phosphorous Content (ESDAC v2.0) within the analysis cell (1 ha)

Latvian name: Augsnes virskārtas fosfora apjoms (ESDAC v2.0) analīzes šūnā (1 ha)

Procedure: Directly derived from Soil chemistry. Processed with `egvtools::` ↪ `downscale2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border and `smooth = FALSE` to keep as original values as reasonable (there is bilinear interpolation involved when projecting from 500 m to 100 m resolution of different CRS).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# P ----

egv=downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = "./Geodata/2024/Soils/ESDAC/chemistry/chemistry/P/P.tif",
  out_path      = "./RasterGrids_100m/2024/RAW/",
```

```
  file_name     = "SoilChemistry_ESDAC-P_cell.tif",
  layer_name    = "egv_504",
  fill_gaps     = TRUE,
  smooth        = FALSE,
  plot_result   = TRUE)
egv
```

6.505 SoilChemistry_ESDAC-phH2O_cell

filename: `SoilChemistry_ESDAC-phH2O_cell.tif`

layername: `egv_505`

English name: Average value of Topsoil pH reaction in water (ESDAC v2.0) within the analysis cell (1 ha)

Latvian name: Augsnes virskārtas reakcija (pH) ūdens šķīdumā (ESDAC v2.0) analīzes šūnā (1 ha)

Procedure: Directly derived from Soil chemistry. Processed with `egvtools::` ↪ `downscale2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border and `smooth = FALSE` to keep as original values as reasonable (there is bilinear interpolation involved when projecting from 500 m to 100 m resolution of different CRS).

```
# libs ----
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}


# pH_H2O ----

egv=downscale2egv(
  template_path = "./Templates/TemplateRasters/LV100m_10km.tif",
  grid_path     = "./Templates/TemplateGrids/tikls1km_sauzeme.parquet",
  rawfile_path  = "./Geodata/2024/Soils/ESDAC/chemistry/chemistry/pH_H2O/pH_
    ↪ H2O.tif",
  out_path      = "./RasterGrids_100m/2024/RAW/",
  file_name     = "SoilChemistry_ESDAC-phH2O_cell.tif",
  layer_name    = "egv_505",
  fill_gaps     = TRUE,
  smooth        = FALSE,
  plot_result   = TRUE)
egv
```

6.506 SoilTexture_Clay_cell

filename: `SoilTexture_Clay_cell.tif`

layername: `egv_506`

English name: Fractional cover of Clay Soils within the analysis cell (1 ha)

Latvian name: Augsnes granulometriskās klases "māls" platības īpatsvars analīzes šūnā (1 ha)

Procedure: Derived from Soil texture product. First, layer is reclassified so that class of interest is 1, other classes are 0. Then processed with `egvtools::input2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# templates ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")

# input ----
combtext=rast("./RasterGrids_10m/2024/SoilTXT_combined.tif")

# EGVs cell ----

# SoilTexture_Clay_cell.tif egv_506

clay10=ifel(combtext==3,1,0)

input2egv(input=clay10,
          egv_template="./Templates/TemplateRasters/LV100m_10km.tif",
          summary_function = "average",
          missing_job = "FillOutput",
          idw_weight = 2,
          outlocation = "./RasterGrids_100m/2024/RAW/",
          outfilename = "SoilTexture_Clay_cell.tif",
          layername="egv_506",
          return_visible = TRUE)
```

6.507 SoilTexture_Clay_r500

filename: `SoilTexture_Clay_r500.tif`

layername: `egv_507`

English name: Fractional cover of Clay Soils within the 0.5 km landscape

Latvian name: Augsnes granulometriskās klases "māls" platības īpatsvars 0,5 km ainavā

Procedure: Derived from SoilTexture_Clay_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Clay"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r500"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Clay_r500.tif egv_507

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r500.tif")
names(slanis)="egv_507"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r500.tif",
            overwrite=TRUE)
```

6.508 SoilTexture_Clay_r1250

filename: `SoilTexture_Clay_r1250.tif`

layername: `egv_508`

English name: Fractional cover of Clay Soils within the 1.25 km landscape

Latvian name: Augsnes granulometriskās klases "māls" platības īpatsvars 1,25 km ainavā

Procedure: Derived from SoilTexture_Clay_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Clay"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r1250"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)


# SoilTexture_Clay_r1250.tif    egv_508

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r1250.tif")
names(slanis)="egv_508"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r1250.tif",
            overwrite=TRUE)
```

6.509 SoilTexture_Clay_r3000

filename: `SoilTexture_Clay_r3000.tif`

layername: `egv_509`

English name: Fractional cover of Clay Soils within the 3 km landscape

Latvian name: Augsnes granulometriskās klases “māls” platības īpatsvars 3 km ainavā

Procedure: Derived from SoilTexture_Clay_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_cell.tif")
    ,
  layer_prefixes = c("SoilTexture_Clay"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r3000"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Clay_r3000.tif    egv_509

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r3000.tif")
names(slanis)="egv_509"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r3000.tif",
            overwrite=TRUE)
```

6.510 SoilTexture_Clay_r10000

filename: `SoilTexture_Clay_r10000.tif`

layername: `egv_510`

English name: Fractional cover of Clay Soils within the 10 km landscape

Latvian name: Augsnes granulometriskās klases “māls” platības īpatsvars 10 km ainavā

Procedure: Derived from SoilTexture_Clay_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Clay"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r10000"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)


# SoilTexture_Clay_r10000.tif   egv_510

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r10000.tif")
names(slanis)="egv_510"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Clay_r10000.tif",
            overwrite=TRUE)
```

6.511 SoilTexture_Organic_cell

filename: `SoilTexture_Organic_cell.tif`

layername: `egv_511`

English name: Fractional cover of Organic Soils within the analysis cell (1 ha)

Latvian name: Augsnes granulometriskās klases "organiskās augsnes" platības īpatsvars analīzes šūnā (1 ha)

Procedure: Derived from Soil texture product. First, layer is reclassified so that class of interest is 1, other classes are 0. Then processed with `egvtools::input2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# templates ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")

# input ----
combtext=rast("./RasterGrids_10m/2024/SoilTXT_combined.tif")

# EGVs cell ----

# SoilTexture_Organic_cell.tif  egv_511

org10=ifel(combtext==4,1,0)

input2egv(input=org10,
          egv_template="./Templates/TemplateRasters/LV100m_10km.tif",
          summary_function = "average",
          missing_job = "FillOutput",
          idw_weight = 2,
          outlocation = "./RasterGrids_100m/2024/RAW/",
          outfilename = "SoilTexture_Organic_cell.tif",
          layername="egv_511",
          return_visible = TRUE)
```

6.512 SoilTexture_Organic_r500

filename: `SoilTexture_Organic_r500.tif`

layername: `egv_512`

English name: Fractional cover of Organic Soils within the 0.5 km landscape

Latvian name: Augsnes granulometriskās klases "organiskās augsnes" platības īpatsvars 0,5 km ainavā

Procedure: Derived from SoilTexture_Organic_cell. First processed with `egvtools::`
`↪ radius_function()`, then rewritten to ensure layername.

```
# libs ----
```

```
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_cell.
    ↪ tif"),
  layer_prefixes = c("SoilTexture_Organic"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r500"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Organic_r500.tif  egv_512

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r500.tif")
names(slanis)="egv_512"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r500.tif",
            overwrite=TRUE)
```

6.513 SoilTexture_Organic_r1250

filename: `SoilTexture_Organic_r1250.tif`

layername: `egv_513`

English name: Fractional cover of Organic Soils within the 1.25 km landscape

Latvian name: Augsnes granulometriskās klases "organiskās augsnes" platības īpatsvars 1,25 km ainavā

Procedure: Derived from SoilTexture_Organic_cell. First processed with `egvtools::` ↪ `radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
```

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_cell.
    ↪ tif"),
  layer_prefixes = c("SoilTexture_Organic"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r1250"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)


# SoilTexture_Organic_r1250.tif egv_513

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r1250.tif")
names(slanis)="egv_513"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r1250.tif",
            overwrite=TRUE)
```

6.514 SoilTexture_Organic_r3000

filename: `SoilTexture_Organic_r3000.tif`

layername: `egv_514`

English name: Fractional cover of Organic Soils within the 3 km landscape

Latvian name: Augsnes granulometriskās klases “organiskās augsnes” platības īpatsvars 3 km ainavā

Procedure: Derived from SoilTexture_Organic_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
```

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_cell.
    ↪ tif"),
  layer_prefixes = c("SoilTexture_Organic"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r3000"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Organic_r3000.tif egv_514

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r3000.tif")
names(slanis)="egv_514"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r3000.tif",
            overwrite=TRUE)
```

6.515 SoilTexture_Organic_r10000

filename: `SoilTexture_Organic_r10000.tif`

layername: `egv_515`

English name: Fractional cover of Organic Soils within the 10 km landscape

Latvian name: Augsnes granulometriskās klases "organiskās augsnes" platības īpatsvars 10 km ainavā

Procedure: Derived from SoilTexture_Organic_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}
```

```
# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_cell.
    ↪ tif"),
  layer_prefixes = c("SoilTexture_Organic"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r10000"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Organic_r10000.tif    egv_515

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r10000.tif")
names(slanis)="egv_515"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Organic_r10000.tif",
            overwrite=TRUE)
```

6.516 SoilTexture_Sand_cell

filename: `SoilTexture_Sand_cell.tif`

layername: `egv_516`

English name: Fractional cover of Sand Soils within the analysis cell (1 ha)

Latvian name: Augsnes granulometriskās klases "smilts" platības īpatsvars analīzes šūnā (1 ha)

Procedure: Derived from Soil texture product. First, layer is reclassified so that class of interest is 1, other classes are 0. Then processed with `egvtools::input2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
```

```
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# templates ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")

# input ----
combtext=rast("./RasterGrids_10m/2024/SoilTXT_combined.tif")

# EGVs cell ----

# SoilTexture_Sand_cell.tif egv_516

sand10=ifel(combtext==1,1,0)
plot(sand10)

input2egv(input=sand10,
          egv_template="./Templates/TemplateRasters/LV100m_10km.tif",
          summary_function = "average",
          missing_job = "FillOutput",
          idw_weight = 2,
          outlocation = "./RasterGrids_100m/2024/RAW/",
          outfilename = "SoilTexture_Sand_cell.tif",
          layername="egv_516",
          return_visible = TRUE)
```

6.517 SoilTexture_Sand_r500

filename: `SoilTexture_Sand_r500.tif`

layername: `egv_517`

English name: Fractional cover of Sand Soils within the 0.5 km landscape

Latvian name: Augsnes granulometriskās klases “smilts” platības īpatsvars 0,5 km ainavā

Procedure: Derived from SoilTexture_Sand_cell. First processed with `egvtools::` ↪ `radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----
```

```
radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Sand"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r500"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Sand_r500.tif egv_517

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r500.tif")
names(slanis)="egv_517"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r500.tif",
            overwrite=TRUE)
```

6.518 SoilTexture_Sand_r1250

filename: `SoilTexture_Sand_r1250.tif`

layername: `egv_518`

English name: Fractional cover of Sand Soils within the 1.25 km landscape

Latvian name: Augsnes granulometriskās klases “smilts” platības īpatsvars 1,25 km ainavā

Procedure: Derived from SoilTexture_Sand_cell. First processed with `egvtools::` ↪ `radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
```

```
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Sand"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r1250"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Sand_r1250.tif    egv_518

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r1250.tif")
names(slanis)="egv_518"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r1250.tif",
            overwrite=TRUE)
```

6.519 SoilTexture_Sand_r3000

filename: `SoilTexture_Sand_r3000.tif`

layername: `egv_519`

English name: Fractional cover of Sand Soils within the 3 km landscape

Latvian name: Augsnes granulometriskās klases "smilts" platības īpatsvars 3 km ainavā

Procedure: Derived from SoilTexture_Sand_cell. First processed with `egvtools::` ↪ `radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
```

```
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Sand"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r3000"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Sand_r3000.tif    egv_519

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r3000.tif")
names(slanis)="egv_519"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r3000.tif",
            overwrite=TRUE)
```

6.520 SoilTexture_Sand_r10000

filename: `SoilTexture_Sand_r10000.tif`

layername: `egv_520`

English name: Fractional cover of Sand Soils within the 10 km landscape

Latvian name: Augsnes granulometriskās klases “smilts” platības īpatsvars 10 km ainavā

Procedure: Derived from SoilTexture_Sand_cell. First processed with `egvtools::` `↪ radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
```

```
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Sand"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r10000"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Sand_r10000.tif   egv_520

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r10000.tif")
names(slanis)="egv_520"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Sand_r10000.tif",
            overwrite=TRUE)
```

6.521 SoilTexture_Silt_cell

filename: `SoilTexture_Silt_cell.tif`

layername: `egv_521`

English name: Fractional cover of Silt Soils within the analysis cell (1 ha)

Latvian name: Augsnes granulometriskās klases "smilšmāls un mālsmilts" platības īpatsvars analīzes šūnā (1 ha)

Procedure: Derived from Soil texture product. First, layer is reclassified so that class of interest is 1, other classes are 0. Then processed with `egvtools::input2egv()` with `fill gaps = TRUE` performing inverse distance weighted (power = 2) filling of gaps at the border.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# templates ----
template10=rast("./Templates/TemplateRasters/LV10m_10km.tif")
template100=rast("./Templates/TemplateRasters/LV100m_10km.tif")
```

```
# input ----
combtext=rast("./RasterGrids_10m/2024/SoilTXT_combined.tif")

# EGVs cell ----

# SoilTexture_Silt_cell.tif egv_521

silt10=ifel(combtext==2,1,0)

input2egv(input=silt10,
          egv_template="./Templates/TemplateRasters/LV100m_10km.tif",
          summary_function = "average",
          missing_job = "FillOutput",
          idw_weight = 2,
          outlocation = "./RasterGrids_100m/2024/RAW/",
          outfilename = "SoilTexture_Silt_cell.tif",
          layername="egv_521",
          return_visible = TRUE)
```

6.522 SoilTexture_Silt_r500

filename: `SoilTexture_Silt_r500.tif`

layername: `egv_522`

English name: Fractional cover of Silt Soils within the 0.5 km landscape

Latvian name: Augsnes granulometriskās klases “smilšmāls un mālsmilts” platības īpatsvars 0,5 km ainavā

Procedure: Derived from SoilTexture_Silt_cell. First processed with `egvtools::` ↪ `radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Silt"),
```

```
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
  radii          = c("r500"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Silt_r500.tif egv_522

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r500.tif")
names(slanis)="egv_522"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r500.tif",
            overwrite=TRUE)
```

6.523 SoilTexture_Silt_r1250

filename: `SoilTexture_Silt_r1250.tif`

layername: `egv_523`

English name: Fractional cover of Silt Soils within the 1.25 km landscape

Latvian name: Augsnes granulometriskās klases "smilšmāls un mālsmilts" platības īpatsvars 1,25 km ainavā

Procedure: Derived from SoilTexture_Silt_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Silt"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
```

```
  n_workers      = 5,
  radii          = c("r1250"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Silt_r1250.tif    egv_523

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r1250.tif")
names(slanis)="egv_523"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r1250.tif",
            overwrite=TRUE)
```

6.524 SoilTexture_Silt_r3000

filename: SoilTexture_Silt_r3000.tif

layername: egv_524

English name: Fractional cover of Silt Soils within the 3 km landscape

Latvian name: Augsnes granulometriskās klases "smilšmāls un mālsmilts" platības īpatsvars 3 km ainavā

Procedure: Derived from SoilTexture_Silt_cell. First processed with egvtools::
↪ radius_function(), then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path   = "./Templates/TemplateGrids/tiles/",
  radii_path      = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path   = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path   = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers    = c("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Silt"),
  output_dir      = "./RasterGrids_100m/2024/RAW/",
  n_workers       = 5,
```

```
  radii           = c("r3000"),
  radius_mode     = "sparse",
  extract_fun     = "mean",
  fill_missing    = TRUE,
  IDW_weight      = 2,
  future_max_size = 5 * 1024^3)


# SoilTexture_Silt_r3000.tif    egv_524

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r3000.tif")
names(slanis)="egv_524"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r3000.tif",
            overwrite=TRUE)
```

6.525 SoilTexture_Silt_r10000

filename: `SoilTexture_Silt_r10000.tif`

layername: `egv_525`

English name: Fractional cover of Silt Soils within the 10 km landscape

Latvian name: Augsnes granulometriskās klases "smilšmāls un mālsmilts" platības īpatsvars 10 km ainavā

Procedure: Derived from SoilTexture_Silt_cell. First processed with `egvtools::radius_function()`, then rewritten to ensure layername.

```
# libs ----
if(!require(terra)) {install.packages("terra"); require(terra)}
if(!require(egvtools)) {remotes::install_github("aavotins/egvtools"); require
    ↪ (egvtools)}

# EGVs radii ----

radius_function(
  kvadrati_path  = "./Templates/TemplateGrids/tiles/",
  radii_path     = "./Templates/TemplateGridPoints/tiles/",
  tikls100_path  = "./Templates/TemplateGrids/tikls100_sauzeme.parquet",
  template_path  = "./Templates/TemplateRasters/LV100m_10km.tif",
  input_layers   = c("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_cell.tif")
    ↪ ,
  layer_prefixes = c("SoilTexture_Silt"),
  output_dir     = "./RasterGrids_100m/2024/RAW/",
  n_workers      = 5,
```

```
  radii          = c("r10000"),
  radius_mode    = "sparse",
  extract_fun    = "mean",
  fill_missing   = TRUE,
  IDW_weight     = 2,
  future_max_size = 5 * 1024^3)

# SoilTexture_Silt_r10000.tif   egv_525

slanis=rast("./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r10000.tif")
names(slanis)="egv_525"
slanis2=project(slanis,template100)
writeRaster(slanis2,
            "./RasterGrids_100m/2024/RAW/SoilTexture_Silt_r10000.tif",
            overwrite=TRUE)
```

6.526 Terrain_ASL-average_cell

filename: `Terrain_ASL-average_cell.tif`

layername: `egv_526`

English name: Average value of height Above Sea Level (m) within the analysis cell (1 ha)

Latvian name: Augstums virs jūras līmeņa (m) analīzes šūnā (1 ha)

Procedure:

6.527 Terrain_Aspect-average_cell

filename: `Terrain_Aspect-average_cell.tif`

layername: `egv_527`

English name: Average value of Terrain Aspect (degree) within the analysis cell (1 ha)

Latvian name: Nogāzes vidējais vērsuma virziens analīzes šūnā (1 ha)

Procedure:

6.528 Terrain_Aspect-iqr_cell

filename: `Terrain_Aspect-iqr_cell.tif`

layername: `egv_528`

English name: Variability of Terrain Aspect (degree) within the analysis cell (1 ha)

Latvian name: Nogāzes vērsuma variabilitāte analīzes šūnā (1 ha)

Procedure:

6.529 Terrain_DiS-area_cell

filename: `Terrain_DiS-area_cell.tif`

layername: `egv_529`

English name: Fractional cover of Terrain Sinks within the analysis cell (1 ha)

Latvian name: Reljefa depresiju bez virszemes noteces platības īpatsvars analīzes šūnā (1 ha)

Procedure:

6.530 Terrain_DiS-area_r500

filename: `Terrain_DiS-area_r500.tif`

layername: `egv_530`

English name: Fractional cover of Terrain Sinks within the 0.5 km landscape

Latvian name: Reljefa depresiju bez virszemes noteces platības īpatsvars 0,5 km ainavā

Procedure:

6.531 Terrain_DiS-area_r1250

filename: `Terrain_DiS-area_r1250.tif`

layername: `egv_531`

English name: Fractional cover of Terrain Sinks within the 1.25 km landscape

Latvian name: Reljefa depresiju bez virszemes noteces platības īpatsvars 1,25 km ainavā

Procedure:

6.532 Terrain_DiS-area_r3000

filename: `Terrain_DiS-area_r3000.tif`

layername: `egv_532`

English name: Fractional cover of Terrain Sinks within the 3 km landscape

Latvian name: Reljefa depresiju bez virszemes noteces platības īpatsvars 3 km ainavā

Procedure:

6.533 Terrain_DiS-area_r10000

filename: `Terrain_DiS-area_r10000.tif`

layername: `egv_533`

English name: Fractional cover of Terrain Sinks within the 10 km landscape

Latvian name: Reljefa depresiju bez virszemes noteces platības īpatsvars 10 km ainavā

Procedure:

6.534 Terrain_DiS-max_cell

filename: `Terrain_DiS-max_cell.tif`

layername: `egv_534`

English name: Maximum Depth in Terrain Sink within the analysis cell (1 ha)

Latvian name: Reljefa depresiju lielākais dziļums analīzes šūnā (1 ha)

Procedure:

6.535 Terrain_DiS-mean_cell

filename: `Terrain_DiS-mean_cell.tif`

layername: `egv_535`

English name: Average Depth in Terrain Sink within the analysis cell (1 ha)

Latvian name: Reljefa depresiju vidējais dziļums analīzes šūnā (1 ha)

Procedure:

6.536 Terrain_Slope-average_cell

filename: `Terrain_Slope-average_cell.tif`

layername: `egv_536`

English name: Average value of Terrain Slope (degree) within the analysis cell (1 ha)

Latvian name: Nogāzes slīpuma vidējā vērtība analīzes šūnā (1 ha)

Procedure:

6.537 Terrain_Slope-iqr_cell

filename: `Terrain_Slope-iqr_cell.tif`

layername: `egv_537`

English name: Variability of Terrain Slope (degree) within the analysis cell (1 ha)

Latvian name: Nogāzes slīpuma variabilitāte analīzes šūnā (1 ha)

Procedure:

6.538 Terrain_TWI-average_cell

filename: `Terrain_TWI-average_cell.tif`

layername: `egv_538`

English name: Average value of Topographic Wetness Index (TWI) within the analysis cell (1 ha)

Latvian name: Topogrāfiskā mitruma indeksa vidējā vērtība analīzes šūnā (1 ha)

Procedure:

Chapter 7

Data access

This chapter provides access to EGVs described in previous parts

References

Brown, C.F., Brumby, S.P., Guzder-Williams, B., Birch, T., Hyde, S.B., Mazzariello, J., Czerwinski, W., Pasquarella, V.J., Haertel, R., Ilyushchenko, S., Schwehr, K., Weisse, M., Stolle, F., Hanson, C., Guinan, O., Moore, R., Tait, A.M., 2022. Dynamic World, Near real-time global 10 m land use land cover mapping. Scientific Data 9, 251. https://doi.org/10.1038/s41597-022-01307-4

Domisch, S., Amatulli, G., Jetz, W., 2015. Near-global freshwater-specific environmental variables for biodiversity analyses in 1 km resolution. Scientific Data 2:150073, 1–13. https://doi.org/10.1038/sdata.2015.73

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., Moore, R., 2017. Google Earth Engine: Planetary-scale geospatial analysis for everyone. Remote Sensing of Environment 202, 18–27. https://doi.org/10.1016/j.rse.2017.06.031

Hansen, M.C., Potapov, P.V., Moore, R., Hancher, M., Turubanova, S.A., Tyukavina, A., Thau, D., Stehman, S.V., Goetz, S.J., Loveland, T.R., Kommareddy, A., Egorov, A., Chini, L., Justice, C.O., Townshend, J.R.G., 2013. High-resolution Global maps of 21st-century forest cover change. Science 342, 850–853. https://doi.org/10.1126/science.1244693

Hijmans, R.J., Cameron, S.E., Parra, J.L., Jones, P.G., Jarvis, A., 2005. Very high resolution interpolated climate surfaces for global land areas. International Journal of Climatology 25, 1965–1978. https://doi.org/10.1002/joc.1276

Karger, D.N., Conrad, O., Böhner, J., Kawohl, T., Kreft, H., Soria-Auza, R.W., Zimmermann, N.E., Linder, H.P., Kessler, M., 2017. Data Descriptor: Climatologies at high resolution for the earth's land surface areas. Scientific Data 4:170122. https://doi.org/10.1038/sdata.2017.122

Lehner, B., Grill, G., 2013. Global river hydrography and network routing: Baseline data and new approaches to study the world's large river systems. Hydrological Processes 27, 2171–2186. https://doi.org/10.1002/hyp.9740

Lehner, B., Verdin, K., Jarvis, A., 2008. New global hydrography derived from spaceborne elevation data. Eos, Transactions, American Geophysical Union 89, 93–94. https://doi.org/10.1029/2008EO100001

Panagos, P., Liedekerke, M.V., Borrelli, P., Köninger, J., Ballabio, C., Orgiazzi, A., Lugato, E., Liakos, L., Hervas, J., Jones, A., Montanarella, L., 2022. European Soil Data Centre 2.0: Soil data and knowledge in support of the EU policies. European Journal of Soil Science 73, e13315. https://doi.org/10.1111/ejss.13315

Shimada, M., Itoh, T., Motooka, T., Watanabe, M., Shiraishi, T., Thapa, R., Lucas, R.,

2013. New global forest/non-forest maps from ALOS PALSAR data (2007–2010). Remote Sensing of Environment 155, 13–31. https://doi.org/10.1016/j.rse.2014.04.014

Wang, L., Liu, H., 2006. An efficient method for identifying and filling surface depressions in digital elevation models for hydrologic analysis and modelling. International Journal of Geographical Information Science 20, 193–213. https://doi.org/10.1080/13658810500433453