globdiversity
rs-enabled ebvs for terrestrial ecosystems

Algorithm Theoretical Basis Document

Fragmentation

Document Ref:	ATBD-FRAG
Version:	3.3
Contract No:	4000120011/17/I-NB
Creation Date:	2018-05-03
Last Modified:	2020-09-30

Submitted by

ITC University of Twente
University of Zurich UZH
UN environment programme WCMC
University of Tartu
DLR
Wageningen University & Research
Tour du Valat
vito
eodc

This page intentionally left blank

Distribution List

This GlobDiversity document is currently stored at the Dropbox are under the folder Dropbox\03_Tasks\03_Task3\01_ATBD

This document is accessible to all participants of the GlobDiversity project.

Summary

This document refers to the activities of task 2 and 3, including sub packages. This document is a draft and the version 2.0 of the Algorithm Theoretical Baseline Document (ATBD).

Document Release Sheet

Authors:	Rene Jochem (WENR) Michiel van Eupen (WENR) Sander Mücher (WENR) Valentin Ziel (DLR) Moritz Hof (DLR) Uta Heiden (DLR) Ruben Van De Kerchove (VITO)	
Approval (internal)	Claudia Röösli (UZH) Vladimir Wingate (UZH)	
Approval (ESA)	Marc Paganini (ESA)	
Distribution:		

Change Record

Version	Date	Page(s)	Change record	Release
0	2018-05-04		First document draft	0.1
1	21-11-2018		Documentation with delivering Prototype from WEnR to DLR	1.0
2	2020-02-06		PR Meeting	2.0
3	2020-03-29		Update chapter 4, 5 and 6, Formatting	3.1
3	2020-05-20		Update all chapters	3.2
3	2020-09-23		Included upscaling results	3.3
3	2020-09-30		Finalisation	3.4

Applicable and reference documents

The following documents apply to the extent specified herein.

Document ID	Document Title	Issue
AD-1	Project Study Report (PSR)	latest
AD-2		
…		

Terms, Definitions and Abbreviations

ATBD	Algorithm Theoretical Basis Document
BOA	Bottom of Atmosphere
BRF	Bidirectional Reflectance Function
CCC	Canopy Chlorophyll Content (g/m^2)
CI_{green}	Green Chlorophyll Index
$CI_{red\text{-}edge}$	Red-edge Chlorophyll Index
DD	Datt Derivative
EBV	Essential Biodiversity variables
EMS	Electromagnetic Spectrum
LAI	Leaf Area Index
MTCI	MERIS Terrestrial Chlorophyll Index
NIR	Near Infrared
PSR	Project Study Report
RS-enabled EBV	Remote sensing enabled Essential Biodiversity variable
SRVI	Simple Ratio Vegetation Index
TOA	Top of Atmosphere
TOC	Top of Canopy
VI	Vegetation index

Table of Contents

List of Figures

List of Tables

1. Introduction

This Algorithm Theoretical Baseline Document (ATBD) describes all technical issues from the prototyping of the Fragmentation (FRAG) in the context of the Remotely Sensed Essential Biodiversity Variables (RS-enabled EBVs) product of the ESA funded GlobDiversity Project. This document shall specify the process flow of the prototyped algorithm and the associated program in more detail.

GlobDiversity is the first large-scale project explicitly designed to develop and engineer RS-enabled EBVs. This project initiated and funded by the European Space Agency (ESA) supports the efforts of the Convention on Biological Diversity (CBD) and Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES), among others, and is adopted under the umbrella of the Group on Earth Observations Biodiversity Observation Network (GEO BON). The GlobDiversity project shall support the initiative to build a global knowledge of biological diversity of terrestrial ecosystems (= on land) and of relevance for society.

There are three RS-enabled EBVs designed as part of the GlobDiversity project with each algorithm documented by such an ATBD:

- Fragmentation (Wageningen Environmental Research WEnR, Wageningen University & Research)
- Canopy chlorophyll concentration (Faculty of Geo-Information Science and Earth Observation ITC, University of Twente)
- Land surface phenology (Dept. of Geography, University of Zurich (UZH), the hereby documented algorithm)

Within the project, these three variables were investigated in detail to contribute to an observation system to assess the variable in an efficient and effective way, covering extensive areas at a fine spatial and temporal resolution. The definition and selection, name and definition of the three RS-enabled EBVs was based on the expertises existing within the project consortium and independent from any efforts of defining and prioritising possible candidate EBVs and RS-enabled EBVs that might have existed at the time of the project's start in 2018.

In the following, the algorithm of the processing chain to derive Fragmentation (FRAG) is described in detail. The algorithm was chosen and developed by the Wageningen University & Research and then transmitted to the German Aerospace Center (DLR) to be translated into a code suitable for cloud computing of larger areas of interest. The algorithm has been chosen and developed with the goal of a potential future global application and with a computational efficient implementation. The ATBD includes a description of the necessary pre-processing steps and the processing step of the core algorithm. In addition, results from the project performed on few test sites globally distributed are presented. In addition, the last chapter presents restrictions of the current implementation and modifications that might be necessary for a potential global processing.

The organization of this document is structured in 8 chapters as shown in the table below.

	Explanation
Chapter 1	Provides an introduction
Chapter 2	Describes the scientific background, and addresses the current standard processing schemes
Chapter 3	Provides information about the input data
Chapter 4	Includes the algorithms of the proposed processing
Chapter 5	Provides information about the product
Chapter 6	Discusses practical considerations for implementation
Chapter 7	Includes the references
Chapter 8	Includes an appendix

2. Scientific background

Habitat loss and fragmentation are and will continue to be one of the major threats to biodiversity (Hanski, 2011; Pereira et al., 2010). Natural habitats in most parts of the world continue to decline in extent and integrity, causing increased fragmentation and loss of sustainable species populations, although there has been significant progress to reduce this trend in some regions and habitats. Not only areas of natural habitats will be lost, but the remaining habitats will become smaller and more isolated (Fahrig, 2003; Opdam, 1991). Construction of transport infrastructure through natural landscapes will also contribute to a further fragmented landscape, especially with a large impact for ground dwelling species (Forman & Alexander, 1998; Jaeger, 2000). Reducing the rate of habitat loss and fragmentation, and eventually halting it, is essential to protect biodiversity and to maintain the ecosystem services vital to human wellbeing (Aichi Targets). Fragmentation, next to ecosystem distribution, land cover and vegetation height are strongly related to the Essential Biodiversity Variable 'Ecosystem structure' or habitat structure (Pereira et al. 2013; Skidmore et al., 2015). Monitoring EBV Ecosystem structure can be supported by remote sensing by amongst others the collection of information on the spatial distribution of habitats and associated land cover and how fragmented these targeted areas are and in the end what it does mean for the species occurring in those habitats. For many species-groups fragmentation can have a large impact on the persistence of the group (Nilsson, Reidy, Dynesius, & Revenga, 2005; Ouborg, 1993; Schipper et al., 2008; Thomas & Hanski, 1997; Villard, Trzcinski, & Merriam, 1999).

The impact of fragmentation on species persistence will occur when the habitat covers roughly less than 20% of the landscape (Rybicki & Hanski, 2013). In cultural landscapes where natural habitat is highly fragmented, any method for assessment of population persistence or potential for biodiversity should be based upon metapopulation theory, taking into account the spatial and temporal dynamics of species (Verboom et al 2001), see Figure 1. Verboom et al (2001) argues that methods based upon species distribution data, population viability analyses (PVA), or landscape indices alone all have severe flaws. Vos et al. (2001) suggested that carrying capacity of habitat areas together with the fragmentation distance of non-habitat areas are good predictors of the proportion of patches in networks occupied by a species.

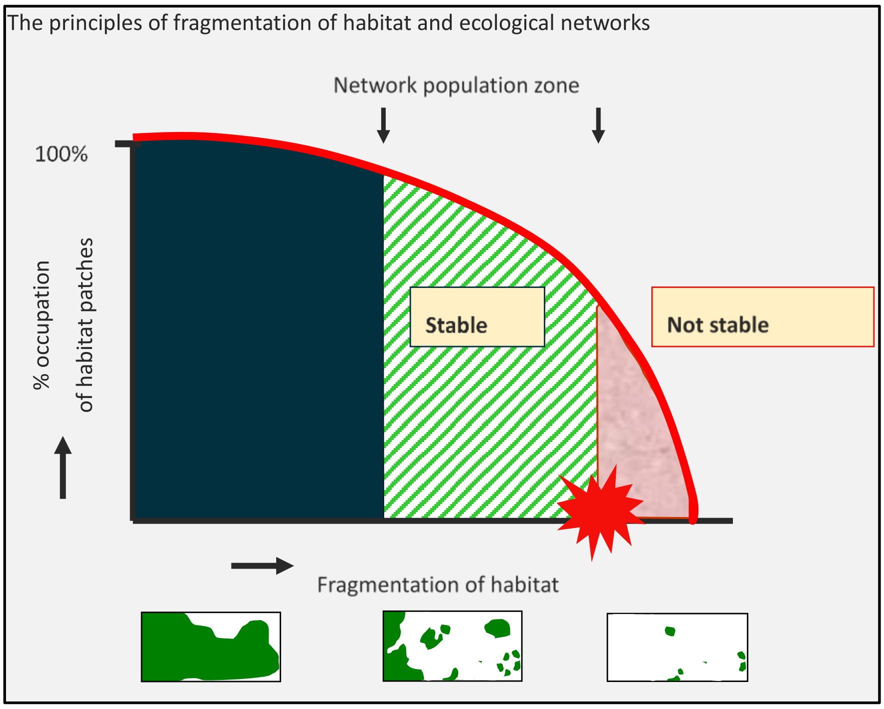

Figure 1: Verboom et al 2001 principles of fragmentation for biodiversity in ecological networks.

In general species persistence depends on four spatial characteristics of landscapes: 1) the size of habitat patches, 2) number of habitat patches, 3) quality of habitat patches and 4) the connectivity between the patches (Hodgson, Moilanen, Wintle, & Thomas, 2011; Opdam, Verboom, & Pouwels, 2003), see Figure 2. Margules and Pressey (2000) as well as Hodgson et al. (2011) conclude that the size and quality of habitats within these large patches should be the focus for nature conservation. However, in urbanised areas restoring habitat connectivity is still one of the main policies to counteract the impact of fragmentation due to infrastructure (Van Der Grift & Pouwels, 2006; C.C. Vos, Opdam, Steingröver, & Reijnen, 2007). Construction of road crossing structures has become a worldwide policy and many species use them to cross roads. However, the impact of these structures on the persistence of endangered species is unknown (Taylor & Goldingay, 2010).

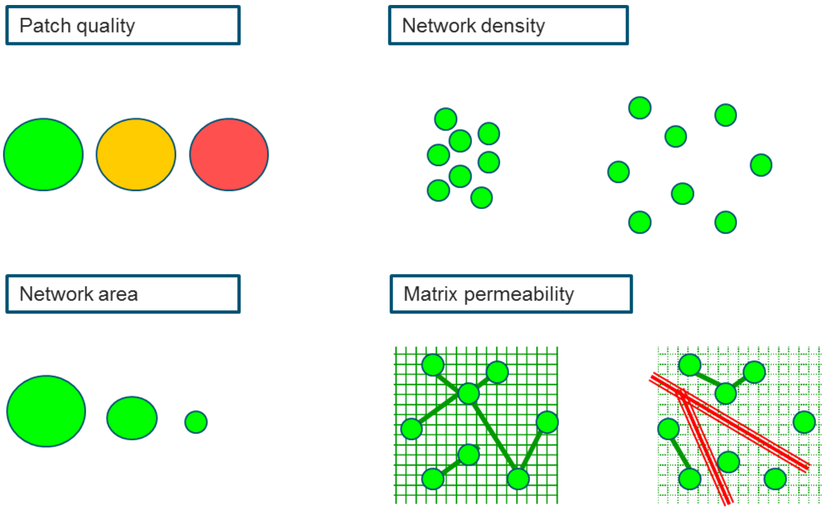

Figure 2: Opdam et al (2003), Landscape configuration related to fragmentation.

The RS-enabled EBV fragmentation to be tested focuses on the effects of fragmentation distances on persistence of organisms across habitat types. This can be calculated as a stack of spatial-temporal metrics based on multiple dispersal distances and (remotely derived) land-cover or habitat types. The spatial resolution will vary by land cover product, with a targeted temporal resolution of one year. The results are individual maps with spatial cohesion values per class, which can be combined to final species cohesion maps by adding up the relevant habitat classes for the species of interest. Fragmentation will be calculated over various distances, e.g.: 500m, 1000m, 5000m and 10000m, depending on the dispersal distances of the targeted species groups

Based on the ESA's selection criteria of globally measurable, ecological meaning, robust and scale free, error estimation, and representativeness of the RS-enabled EBV and our extensive literature review (described in detail in the PSR), the RS-enabled EBV-Fragmentation development of products focuses here on implementing the Hanski algorithm with the LARCH-SCAN using classified land cover data from satellite imagery. LARCH-SCAN is a spatial model that has been developed by WENR and has been used in fragmentation studies since 1998 (Foppen & Chardon 1998, Foppen 1999). Since then the models follow the same principle[1]. The LARCH-SCAN model determines the Spatial Cohesion of nature areas. Spatial cohesion looks at the spatial configuration of targeted habitats/land cover types and what the spatial distances are in between these targeted areas that species have to travel to maintain a sustainable population. For each cell the amount of habitat in its surrounding is determined. Habitat further away is accounted for less than habitat close by, using Hanski's (1994) negative exponential function for cohesion (e-αd; α being the species specific dispersal capacity and d the distance between cells or patches) The output is a relative measure for spatial cohesion of the suitable habitat for a species or an ecosystem. This measure provides the opposite value for fragmentation and has been used to determine promising areas for species and connections between these areas (Sluis & Chardon, 2001; Groot Bruinderink et al., 2003).

It is foreseen to provide the user with a standard set of ecoprofiles applicable at global/continental scale. An ecoprofile can be defined as a set of species demanding similar dimensions of ecosystem coherence in order to persist at a regional scale (Figure 3). "Similar" is meant here in a relative sense, and refers to the similarity in choice of:

- required ecosystem type(s),
- area requirements,
- and dispersal capacity of the species

encompassed by a single ecoprofile, relative to the difference between species classified in other ecoprofiles.

[1] An additional function for the model, LARCH-SCAN with resistance, has been developed. This functions can take barrier effects and resistance of the landscape into account. The results of this LARCH-SCAN module have be used as to assess the impact of infrastructure on the viability of ecological networks (Van der Grift & Pouwels 2006). For the input of Globdiversity this function is not used.

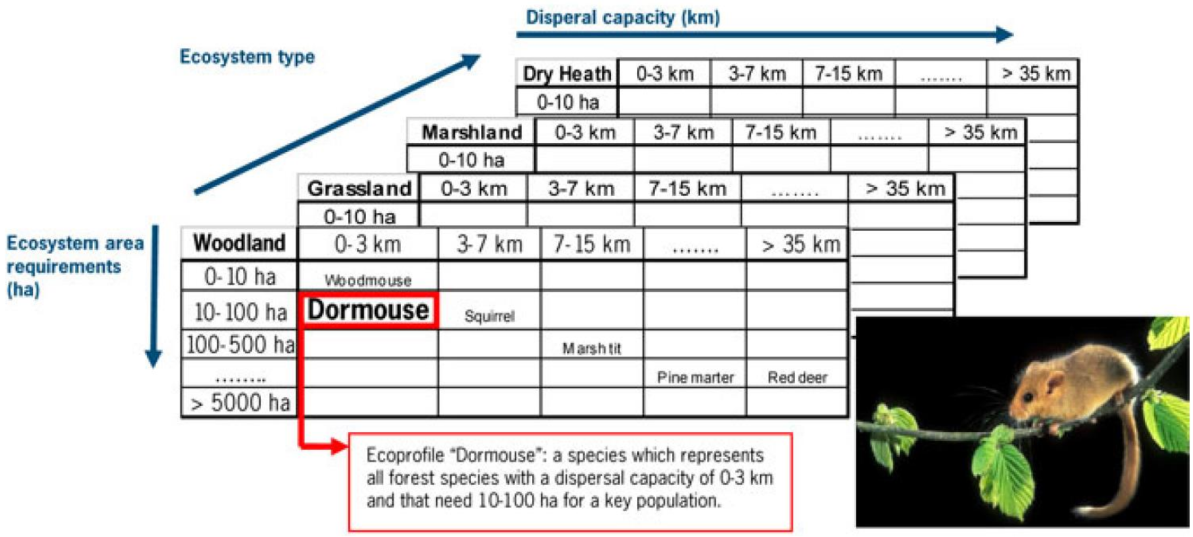

Figure 3: Design of three-dimensional ecoprofile matrices, one per identified ecosystem type, based on the carrying capacity of regional ecosystems (vertical axis), and the inter-patch distance that can be crossed during dispersal. Species are assigned to cells in the matrix by their habitat preference, individual habitat area requirements, and dispersal capacity. Each cell in the matrices represents one ecological profile. (Opdam et al., 2008).

The elaborated ecoprofile described in Chapter 4 ("Black Rhino") is based on parameters derived from literature as described in Emslie (2102), Polz et al (2014), Opdam et al (2008), Lent & Fike (2003) and Linklater & Hutcheson (2010). As habitat is selected all shrubland and open forest classes from the used land cover map (See chapter 3.1). As dispersal capacity a distance of 10 km is used to account for both the large homerange sizes (Polz et al., 2014; Lent & Fike, 2003) and the relative weak (re)colonization response capacity (Linklater & Hutcheson, 2010). No threshold is used for the size of a key population, expressing the results as the total hectares of habitat in connected clusters of habitat.

3. Input Data

Within the GlobDiversity project, two versions for the FRAG algorithms have been developed and tested. The version that can be run on Windows operating systems is a full version including all four steps described in chapter 4.2 and is labelled with *WIN*. Another version that can be run on Linux operating systems has been prototyped by DLR. It includes step 01 (Python based split code) and step 02 (C++ code) (see section 4.2) that are run-time intensive. This version is labelled with *LINUX*. According to the two version, input data differ and are described in the following sections.

3.1. Input image types

Required input for the EBV fragmentation is a raster map that provides information on the habitat of a species in terms of ecosystem -, habitat - or land cover types. In a pragmatic approach all classified global and regional (land cover) products can be used to derive basic habitat types of species as the input to derive fragmentation indices.

Since the EBV Fragmentation is scale and resolution free it is possible to use and compare numerous land cover products from several sources at multiple scales.

The following data set are used for the example processing steps of the test site Kruger Park. These processing steps are valid as well as for the global processing steps.

Original land cover data input from WEnR before prototyping:

- Dynamic Land Cover - 100m Africa V1 GEOTIFF: This 100m Global Land Cover version 1 products are provided as a set of single-band GeoTIFF files. Layer used: LCCS: Discrete map according FAO Land Cover Classification System. https://land.copernicus.eu/¬global/products/lc (Figure 4).

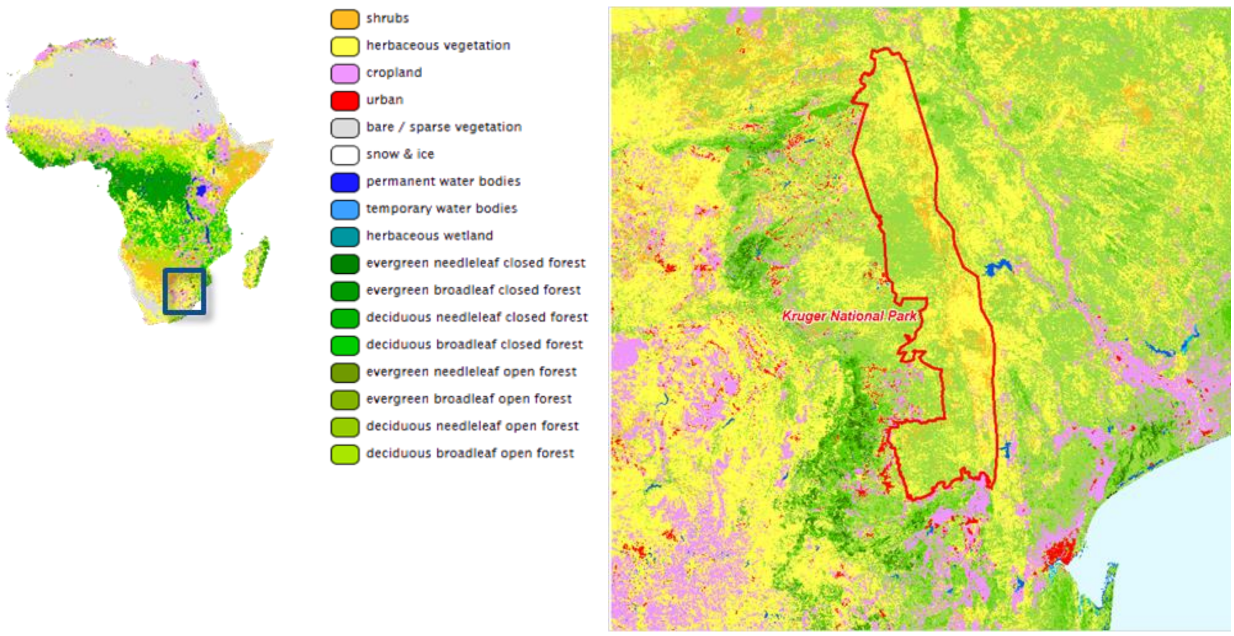

Figure 4: PROBA-V based LCCS Land cover V1 for Africa with the defined classes. Zoom: Globdiversity test site area Kruger Park, South Africa.

Land cover data input for processing during prototyping and use case demonstration of the ESA GlobDiversity project:

- Finer Resolution Observation and Monitoring of Global Land Cover for all test sites with the resolution of the given Sentinel 2 granules with a GDAL/Rasterio processing step (Gong P., et al., 2019), FROM-GLC10 with 10 meters resolution as GeoTIFF files. Classification system.

3.2. Image preprocessing

Spatial location and extend of general habitat types can be selected from remotely sensed land cover products, directly or by combining them though pre-processing. For large regional or continental studies such as Africa where the focus is on non-specific large scale ecosystems (e.g. "forest" fragmentation) products like the mentioned PROBA-V 100 meter land cover datasets can be used directly. For more local (or more habitat specific) studies such as the Camargue pilot area, site specific land cover or habitat maps, or alternative 20 meter Sentinel-2 derived land cover can be used and/or created.

a) Original input information by WEnR for the *WIN* version

As shown in Table 1 the spatial reference of the PROBA-V based LCCS Land cover V1 for Africa is given in a geographic coordinate system (GCS_WGS_1984). To perform reliable area calculations a projected coordinate system is needed, so the maps was given an WGS_1984_World_Mercator projection coordinate system[2] (which has the same GCS_WGS_1984 Geographic Coordinate System), setting the original cell size of 0,00099206349 decimal degrees into 115.355 meters for Kruger Park.

Table 1: Properties of land cover input file c_gls_LC100-LCCS_201501010000_AFRI_PROBAV_1.0.1.tiff

Columns_and_Rows	90720, 80640
Number_of_Bands	1
Cell_Size_X._Y (decimal degrees)	0,00099206349, 0,00099206349 (+/- 100m on the equator)
Uncompressed_Size	6,81 GB
Format	TIFF
Source_Type	Thematic
Pixel_Type	unsigned integer
Pixel_Depth	8 Bit
NoData_Value	255
Colormap	Present

[2] Some of the more commonly used spatial reference systems are: 4326 - WGS 84 Long Lat, 4269 - NAD 83 Long Lat, 3395 - WGS 84 World Mercator, 2163 - US National Atlas Equal Area, Spatial reference systems also exist for each NAD 83, WGS 84 UTM zone - UTM zones are one of the most ideal for measurement, but only cover 6-degree regions (http://spatialreference.org)

Compression	LZW Compression
Extent	Top 45,0004960317, Left -30,0004960317, Right 59,9995039683, Bottom -34,9995039683
Spatial_Reference, XY_Coordinate_System	GCS_WGS_1984 WKID: 4326 Authority: EPSG Angular Unit: Degree (0,0174532925199433) Prime Meridian: Greenwich (0,0) Datum: D_WGS_1984 Spheroid: WGS_1984 Semimajor Axis: 6378137,0 Semiminor Axis: 6356752,314245179 Inverse Flattening: 298,257223563
Used Spatial_Reference, Projected_Coordinate_System	WGS_1984_World_Mercator WKID: 3395 Authority: EPSG Projection: Mercator False_Easting: 0.0 False_Northing: 0.0 Central_Meridian: 0.0 Standard_Parallel_1: 0.0 Linear Unit: Meter (1.0) Geographic Coordinate System: GCS_WGS_1984 Angular Unit: Degree (0,0174532925199433) Prime Meridian: Greenwich (0,0) Datum: D_WGS_1984 Spheroid: WGS_1984 Semimajor Axis: 6378137,0 Semiminor Axis: 6356752,314245179 Inverse Flattening: 298,257223563

Degrees of latitude are parallel so the distance between each degree remains almost constant but since degrees of longitude are farthest apart at the equator and converge at the poles, their distance varies greatly. The range in degree of latitude varies slightly (due to the earth's slightly ellipsoid shape) from 110.567 km at the equator to 111.699 km at the poles. A degree of longitude is widest at the equator at 111.321km and gradually shrinks to zero at the poles. The used World Mercator projection preserves angles locally, implying that local shapes are not distorted. Also, at any given point, local scale is constant in all directions (Snyder, 1987).

Another choice is to run LARCH-Scan with the given/original geographic coordinate system (e.g. GCS_WGS_1984). The distances and the derived parameters in LARCH scan then needs to be converted using the distance of Latitude in meters at the equator. See paragraph 4.2.2 for an example for the Kruger park test site, in both meters and decimal degrees. For any area calculation a projected coordinate system needs to be used (Snyder, 1987). Thus, to express the fragmentation results as “the total hectares of habitat in connected clusters of habitat” a projected coordinate system needs to be chosen. When a geographic coordinate system is used (e.g. GCS_WGS_1984) the output of the clusters should only be expressed for larger areas on the globe as “the total number of cells of habitat in connected clusters of habitat” (See also 4.3.5).

b) Input used for prototyping by DLR for the *LINUX* version

A high-resolution (10m) global land cover (GLC) product for the year 2017 have been generated by Gong et al., (2019), and made freely and openly available. For the prototyping phase the complete global data set was downloaded from the University of Tsinghua. In the next step a virtual, global scene was created using GDAL VRT, which serves as the basis for the following steps. The VRT driver is a format driver for GDAL that allows a virtual GDAL dataset to be composed from other GDAL datasets or image files with repositioning, and algorithms potentially applied as well as various kinds of metadata altered or added. VRT descriptions of datasets can be saved in an XML format normally given the extension *.vrt.

The global landcover VRT file was cut out using the Sentinel 2 geometries of the test sites and saved again as a GeoTIFF file (see Figure 5 and Figure 6). The ten GLC classes of this product are saved in one band as in the original input example from WEnR before. The equivalent of the example scene Kruger Park corresponds to one Sentinel 2 data granule with the designation 36JUT and is used for further processing explanation (details see Table 2). Due to the high resolution it makes little sense to create an even larger map section, since the computing time increases extremely strongly here and one would have to switch back to subtiling or memory core processing or optimizations like with CUDA.

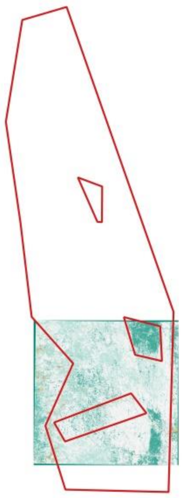

Figure 5: Cropped Landcover scene from Sentinel 2 36JUT granule with extend of the full test site

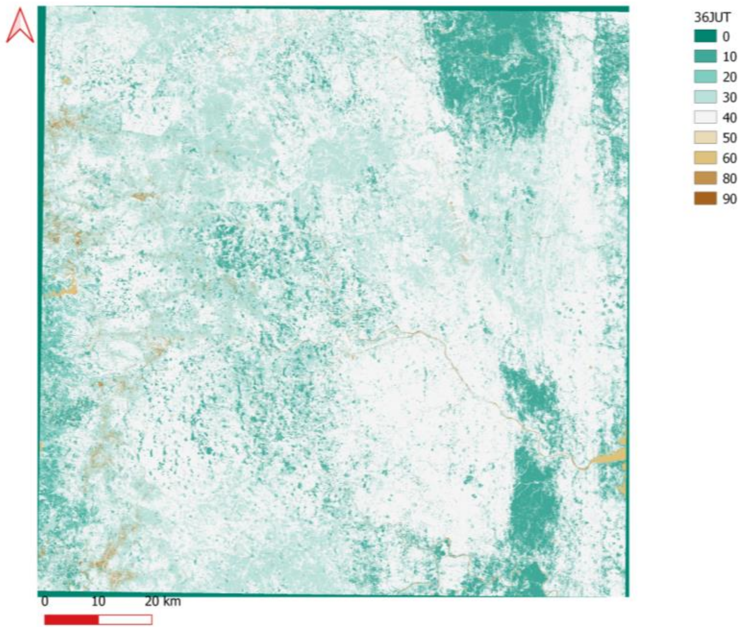

Figure 6: The same Landcover Scene 36JUT with colored classes and 10 meter resolution per pixel.

Table 2: Properties of land cover input file 36JUT.tif

File	36JUT.tif
Columns_and_Rows	11096, 11095
Number_of_Bands	1
Pixel size	11.020043882637973, -11.020043882637973
Uncompressed_Size	234 MB

Format	GTiff/GeoTIFF
Source_Type	Thematic
Pixel_Type	unsigned integer
Pixel_Depth	16 Bit
NoData_Value	255
Colormap	Gray
Compression	LZW Compression
Origin	3452232.884813876822591, -2784651.462106858380139
Corner Coordinates	Upper Left (3452232.885,-2784651.462) (31d 0'42.97"E, 24d24' 2.09"S) Lower Left (3452232.885,-2906918.849) (31d 0'42.97"E, 25d24' 8.36"S) Upper Right (3574511.292,-2784651.462) (32d 6'37.37"E, 24d24' 2.09"S) Lower Right (3574511.292,-2906918.849) (32d 6'37.37"E, 25d24' 8.36"S) Center (3513372.088,-2845785.156) (31d33'40.17"E, 24d54' 8.93"S)
Used Spatial_Reference, Projected_Coordinate_System	WGS_1984_World_Mercator WKID: 3395 Authority: EPSG Projection: Mercator False_Easting: 0.0 False_Northing: 0.0 Scale factor at natural origin: 1 Latitude of natural origin: 0.0 Longitude of natural origin: 0.0 Linear Unit: Meter (1.0) Block=11096x1 INTERLEAVE=BAND Geographic Coordinate System: GCS_WGS_1984 Angular Unit: Degree (0,0174532925199433) Prime Meridian: Greenwich (0,0) Datum: D_WGS_1984 Spheroid: WGS_1984

In the next step the classes were extracted fully automatically with a Python script and saved in individual files (Split algorithm). The file was saved as raster file with the GDAL Driver EHdr as so-called FLT file. A FLT floating point raster file is a binary file with floating point values representing raster data. The necessary files with headers and projection information are generated and stored in the same folder and with the same naming convention as well. There is a maximum of 11 classes in total and thus the data volume multiplies by a factor of ten to eleven, depending on the given scene. The split program recognizes the number of classes fully automatically and only saves the existing classes. See Figure 7for an example of the 36JUT-scene, showing a single value as .flt file. Since the next program in the processing chain (LARCH SCAN) requires the appropriate input, all class scenes have the data type ‘float32’ and must be binary and uncompressed. Therefore, it’s essential that the header file corresponding to the class images exists.

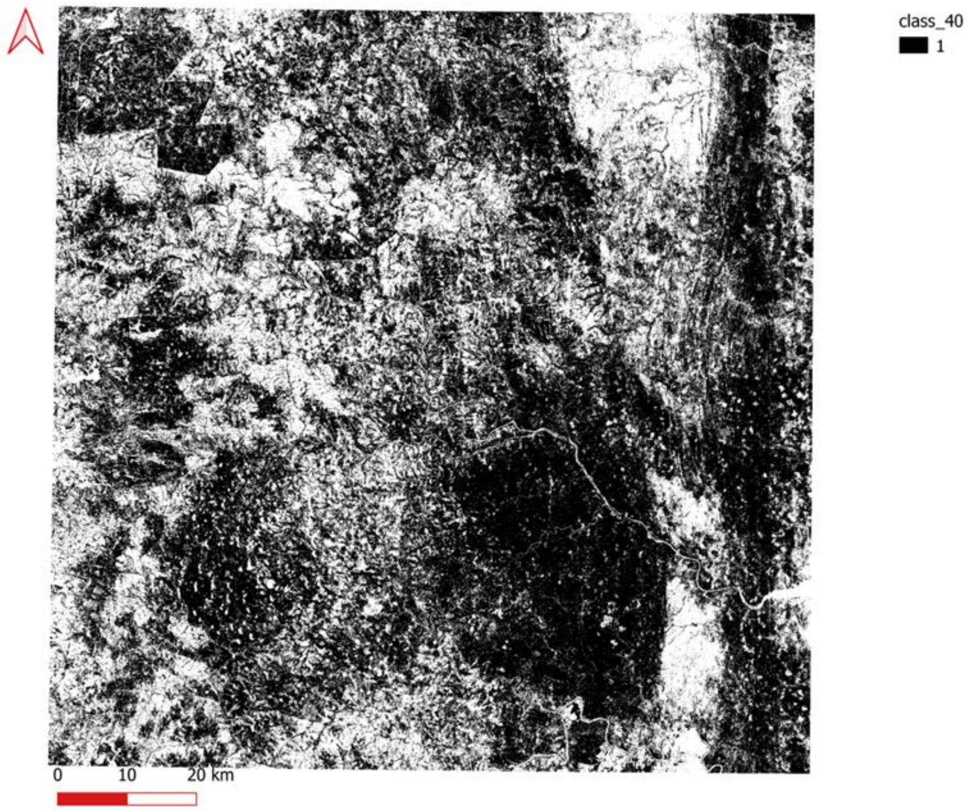

Figure 7: Derived Landcover scene 36JUT from class 40 with single value as .flt file

The FROM GLC10 input data was previously re-projected into the World Mercator projection (https://spatialreference.org/ref/epsg/wgs-84-world-mercator/) using GDAL, since it was using WGS84 by default. When using distance/travel related metrics it's preferred to use a projected coordinated system, since ecoprofiles distances are known and given in SI base units (meters, kilometers). The Mercator projection is the standard map projection for many major online street mapping and navigation services because of its unique property of representing any course in any direction as a straight line, while distortions at a regional scale are limited, except for the poles. Technically the LARCH SCAN program can handle any geographic or projected coordinate system, as long as the corresponding ecoprofile-distance is expressed in meters and related to the cell-size of the used dataset expressed in meters (e.g. expressing decimal degrees in meters without reprojecting, see par. 5.2.2). The LARCH SCAN program works sequentially and calculates one class image after the other with the corresponding parameters from the next chapter (percentage and alpha value).

4. Algorithm description

4.1. Theoretical description

This part of the Algorithmic Theoretical Basis Document (ATBD) describes the algorithm proposed to produce global and regional fragmentation maps of terrestrial ecosystems from remotely sensed land cover products. After a thorough literature review, the LARCH-Scan Hanski metric was found to be an appropriate metric to be calculated as a RS-enabled EBV. Other ways of expressing the EBV could possibly be added, but are not described in this document.

LARCH-SCAN is a spatial model that has been developed by WENR and has been used in fragmentation studies since 1998 (Foppen & Chardon 1998, Foppen 1999). Since then the principles of the model has not been changed[3]. LARCH-SCAN determines the Spatial Cohesion of habitat areas. For each cell the amount of habitat in its surrounding is determined. Habitat further away is accounted for less than habitat close by, using Hanski's (1994) negative exponential function for cohesion ($e^{-\alpha d}$; α being the species specific dispersal capacity and *d* the distance between cells or patches), see Figure 8. The output is a relative measure for spatial cohesion of the suitable habitat for a species or an ecosystem. This measure provides the opposite value for fragmentation and has been used to determine promising areas for species and connections between these areas (Sluis & Chardon, 2001; Groot Bruinderink et al., 2003).

Das verknüpfte Bild kann nicht angezeigt werden. Möglicherweise wurde die Datei verschoben, umbenannt oder gelöscht. Stellen Sie sicher, dass die Verknüpfung auf die korrekte Datei und den korrekten Speicherort zeigt.

$$SC_i = \sum RU_j \cdot e^{-\alpha \cdot d_{ij}}$$

d_{ij} is the distance between the contributing cell j and cell i (meters)

RU_j is the maximum number of reproductive units in cell j

Figure 8: Conceptual scheme of the LARCH-SCAN model. The connectivity is based on the contribution of all habitat grid cells in the surrounding of a targeted grid cell. The contribution to a cell is based on the habitat quality of the surrounding cells and the distance to these cells, which reduces the contribution by a negative exponential function. Large trees in the figure represent grid cell with high suitability values (peak of the curves below). LARCH-SCAN calculates the sum of all contributions for all grid cells in the map.

[3] An additional function for the model, LARCH-SCAN with resistance, has been developed. This functions can take barrier effects and resistance of the landscape into account. The results of this LARCH-SCAN module have be used as to assess the impact of infrastructure on the viability of ecological networks (Van der Grift & Pouwels 2006). For the input of GlobDiversity this function is not used.

Optionally the level of fragmentation can be used to derive clusters of (well) connected habitat cells to express fragmentation in terms of cluster size. Cohesion maps can be used to derive clusters of connected habitats cells to construct ecological networks Patterns of cohesion values can be used for planning corridors between local patches or to improve weaker spots in networks. Depending on the application, different thresholds can be set to create the clusters. Thresholds usually range from:

- 0 = no cohesion;
- < 0.1 = weak cohesion;
- 0.1-0.5 = higher cohesion values, usually corridors
- to >0.5 strong cohesion forming networks

4.2. Work flow of product generation

This paragraph describes the proposed workflow, used both for the test sites as global application. Basic principles used from the generation of the workflow are that:

- The end-user should have maximum flexibility in the generation of the final product keeping the process as generic as possible over the total workflow.
- The end-user is interested in the generation of a fast and simple end-product, not in the technical calculation procedure which lies behind.
- Preventing multiple, time consuming, calculations of the same fragmentation raster for each input product using a “stack” pre-calculated fragmentation raster for each class of the input map
- The workflow should be identical with all (suitable) input data possibly provided by the end-user

Workflow:

The product generated using the benchmarked algorithm involves four steps (see also Figure 9).

Step01: Select the land cover product to start with and run Split Map algorithm for an example script to split the map into separate maps per class or the corresponding source code belonging to this project.

Step02: Run the LARCH-SCAN (Normalised) algorithm for all classes in the map.

Step03: Select the ecoprofile of choice (Flagship species with a habitat/LC-types & fragmentation distance). Flagship species can be symbolic: in this document an example is given for the ecoprofile “Black Rhino”.

Step04: (Optional): Run the cluster algorithm to derive size and number of clusters in the area of interest

Only step 3 (and optionally step 4) has to be re-run for other species / ecosystems of interest based on the same input data

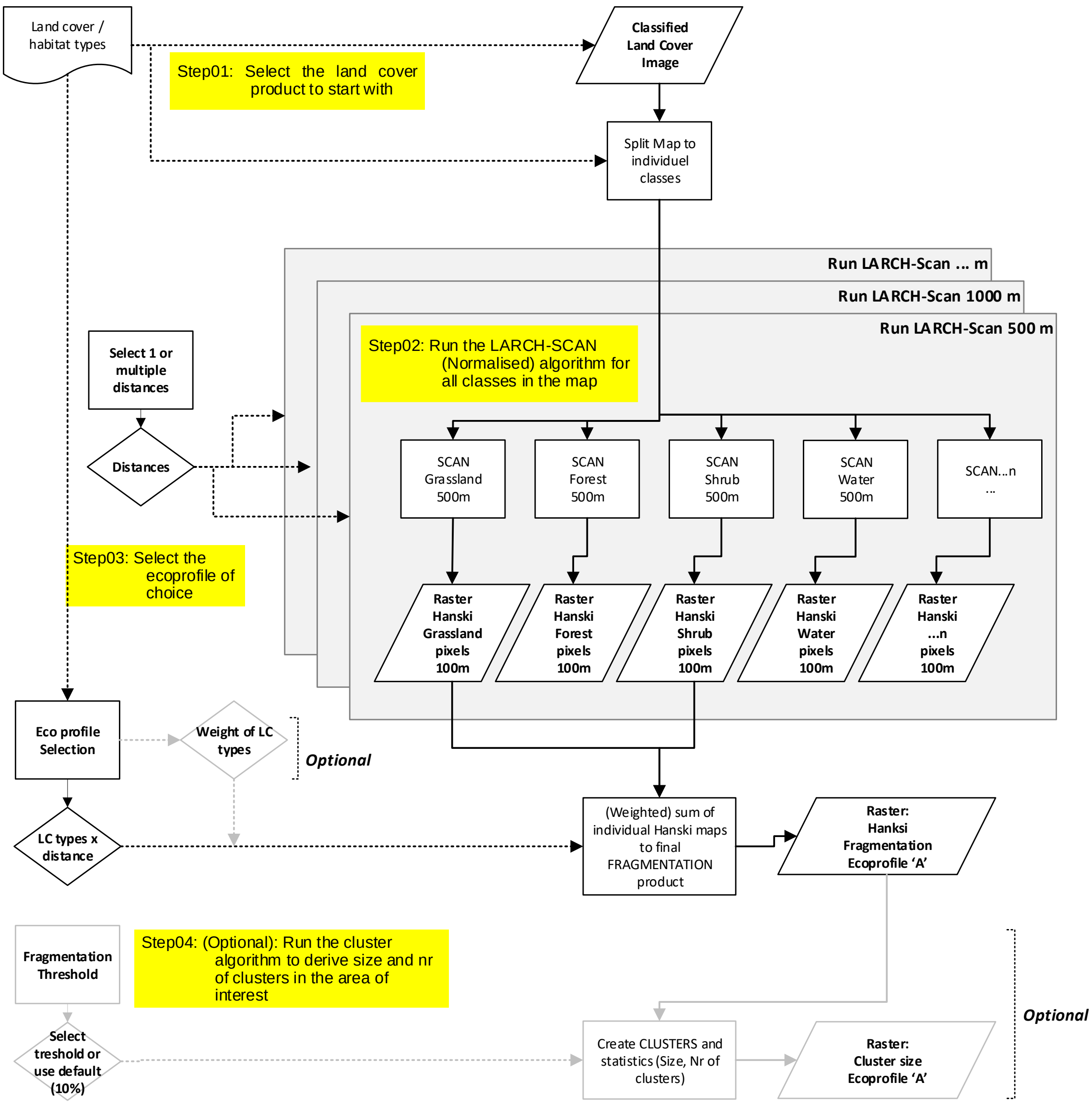

Figure 9: Workflow 1: Proposed workflow creating a stack of fragmentation rasters to be combined in a next step using a user specific ecoprofile: combination of habitat/LC-types x fragmentation distance

4.2.1. Step 01 - Splitting

For the test site Kruger the following land cover types have been used as an input from the PROBA-V LCCS land cover V1 for Africa (see Table 3) before the prototyping.

Table 3: Overview of the land cover types used as input for Kruger NP in step01

LC100 code	In Kruger	Land Cover Class	Definition according UN LCCS	Color code (RGB)
0	Yes	No PROBAV data available		51, 51, 51

111		Closed forest, evergreen needle leaf	tree canopy >70%, almost all needle leaf trees remain green all year. Canopy is never without green foliage.	0, 130, 0
112	Yes	Closed forest, evergreen, broad leaf	tree canopy >70%, almost all broadleaf trees remain green year round. Canopy is never without green foliage.	0, 153, 0
113		Closed forest, deciduous needle leaf	tree canopy >70%, consists of seasonal needle leaf tree communities with an annual cycle of leaf-on and leaf-off periods	0, 179, 0
114	Yes	Closed forest, deciduous broad leaf	tree canopy >70%, consists of seasonal broadleaf tree communities with an annual cycle of leaf-on and leaf-off periods.	0, 204, 0
121		Open forest, evergreen needle leaf	top layer- trees 15-70% and second layer- mixed of shrubs and grassland, almost all needle leaf trees remain green all year. Canopy is never without green foliage.	112, 153, 0
122	Yes	Open forest, evergreen broad leaf	top layer- trees 15-70% and second layer- mixed of shrubs and grassland, almost all broadleaf trees remain green year round. Canopy is never without green foliage.	131, 179, 0
123		Open forest, deciduous needle leaf	top layer- trees 15-70% and second layer- mixed of shrubs and grassland, consists of seasonal needle leaf tree communities with an annual cycle of leaf-on and leaf-off periods	150, 204, 0
124	Yes	Open forest, deciduous broad leaf	top layer- trees 15-70% and second layer- mixed of shrubs and grassland, consists of seasonal broadleaf tree communities with an annual cycle of leaf-on and leaf-off periods.	169, 230, 0
20	Yes	Shrubs	These are woody perennial plants with persistent and woody stems and without any defined main stem being less than 5 m tall. The shrub foliage can be either evergreen or deciduous.	255, 187, 34
30	Yes	Herbaceous vegetation	Plants without persistent stem or shoots above ground and lacking definite firm structure. Tree and shrub cover is less than 10%.	255, 255, 76
40	Yes	Cultivated and managed vegetation/agriculture (cropland)	Lands covered with temporary crops followed by harvest and a bare soil period (e.g., single and multiple cropping systems). Note that perennial woody crops will be classified as the appropriate forest or shrub land cover type.	240, 150, 255
50	Yes	Urban / built up	Land covered by buildings and other man-made structures	255, 0, 0
60	Yes	Bare / sparse vegetation	Lands with exposed soil, sand, or rocks and never has more than 10% vegetated cover during any time of the year	220, 220, 220
70		Snow and Ice	Lands under snow or ice cover throughout the year.	255, 255, 255
80	Yes	Permanent water bodies	lakes, reservoirs, and rivers. Can be either fresh or salt-water bodies.	25, 25, 255
81	Yes	Temporary water bodies		60, 160, 255
90	Yes	Herbaceous wetland	Lands with a permanent mixture of water and herbaceous or woody vegetation. The vegetation can be present in either salt, brackish, or fresh water.	0, 150, 160

200	Yes	Open sea	Oceans, seas. Can be either fresh or salt-water bodies.	0, 0, 128
255		Not classified		0, 0, 0

For the test site Kruger the following land cover types have been used as input from the cropped FROM GLC10 36JUT input granule (see Table 4).

Table 4: Classes and codes of the global land cover (GLC) product as adapted for the generic land cover classes required for the proposed algorithms.

GLC class Code	GLC class name	In Kruger
0	No Data	Yes
10	Cropland	
30	Grassland	
40	Shrubland	
50	Wetland	
70	Tundra	
20	Forest	
60	Water	
80	Impervious surface	
90	Bareland	
100	Snow/Ice	No

The splitting was done with a small Python 3 script which only needs an input file and an output path. Dependencies exist to the libraries gdal, gdal functions, gdalnumeric, argparse, numpy and some system libraries. Due to the detail and length of the programm, no publication is made here. An example output is given in Figure 10 in paragraph 4.3.4.

4.2.2. Step 02 - Run LARCH-Scan covering a broad range of distances

Usual distances for the given spatial resolution of the map (~100m) are (Opdam et al. 2008):

- 500 meter
- 1000 meter
- 5000 meter
- 10000 meter

The values were adopted for the finer resolution of 10 meters for the prototyping phase and were not discussed further as there was a lack of expert knowledge in this area from DLR and a precise analysis is required.

The *WIN* and *LINUX* LARCH-Scan programs assume cell sizes in meters. When using input data in decimal degrees (e.g. GCS_WGS_1984), the distances and the derived parameters in LARCH-Scan needs to be converted using the distance of Latitude in meters (e.g. at the equator). For this reason, the FROM GLC10 input data was previously re-projected into the World Mercator projection using GDAL, since it was using WGS84 by default.

The following example is given for the data used in the Kruger park test site for the PROBA-V based LCCS Land cover in the original GCS_WGS_1984 projection (See Table 5).

Table 5: Distances and the derived parameters in LARCH in meters and Decimal Degrees (DD)

Dm	*Dcell*	*Ddd*	*Alpha_m*	*Alpha_dd*

Fragmentation distance in m	Fragmentation distance in nr. of cells	Fragmentation distance in DD	α (using cellsize in m)	α (using cellsize in DD)
10000	91.166	0.090442899	0.300	33122.913
5000	45.583	0.045221449	0.599	66245.826
1000	9.117	0.009044290	2.996	331229.130
500	4.558	0.004522145	5.991	662458.261
DecDegr_m:	110567	m at the equator		
CellSize_dd:	0.00099206349	DD cellsize		
CellSize_m:	109.6894839	m cellsize	*(CellSize_dd * DD)*	
Dcell = Dm / CellSize_m				
Ddd = Dm / DD				
Alpha_m = (log(100)-log(100-Pct)) / (Dm/1000)			*with Pct = 95% (see par. 5.3)*	
*Alpha_dd = Alpha_m * DecDegr_m*				

4.2.3. Step 03 - Black Rhino ecoprofile: selection of land cover types

Based on literature (Polz et al., 2014; Opdam et al., 2008; Lent & Fike, 2003; Linklater & Hutcheson, 2010) habitat for the Black Rhino is selected from the map. All shrubland and open forest classes are selected from the used land cover map (see Table 6Table 3).

Table 6: Overview of the land cover types used as input for Kruger NP in step03.

LC100 code	Black Rhino habitat	Land Cover Class	Definition according UN LCCS
122	YES	Open forest, evergreen broad leaf	top layer- trees 15-70% and second layer- mixed of shrubs and grassland, almost all broadleaf trees remain green year round. Canopy is never without green foliage.
124	YES	Open forest, deciduous broad leaf	top layer- trees 15-70% and second layer- mixed of shrubs and grassland, consists of seasonal broadleaf tree communities with an annual cycle of leaf-on and leaf-off periods.
20	YES	Shrubs	These are woody perennial plants with persistent and woody stems and without any defined main stem being less than 5 m tall. The shrub foliage can be either evergreen or deciduous.

Selected distances for the Black Rhino ecoprofile: 10000 meter

As described in Chapter 2 a distance of 10 km is used to express the Black Rhino's dispersal capacity. This distance is accounting for both the large home range (Polz et al., 2014; Lent & Fike, 2003) and the relative weak (re)colonization response capacity (Linklater & Hutcheson, 2010).

4.2.4. Step 04 - Run the cluster algorithm to derive size and number of clusters in the area of interest (optional)

The used threshold for the connectivity value >= 0.5, which means a very well connected habitat to form clusters. This is based on the fact that Black Rhino's show a very weak colonization response even when overlapping home ranges are present (Linklater & Hutcheson, 2010).

To express the fragmentation results as "the total hectares of habitat in connected clusters of habitat" a projected coordinate system needs to be used (Snyder, 1987). When a geographic coordinate system is used (e.g. GCS_WGS_1984) the output of the clusters should only be expressed for larger areas on the globe as "the total number of cells of habitat in connected clusters of habitat".

4.3. Algorithm mathematical description

*4.3.1. Split classes Habitat map *WIN* - Step 01*

The following example Python-script shows the algorithm for creating input splitted habitat maps as it is used in ArcMAP (ESRI) by WEnR. For the script code, see Chapter 9.1.

*4.3.2. Split classes Habitat map *LINUX* - Step 01*

The following Python-script contains the code for creating splitted habitat maps as it is used and required for the subsequent LARCH-SCAN *LINUX* step 02. For the script code, see Chapter 9.2.

*4.3.3. LARCH-SCAN (Normalised) - *WIN* - Step 02*

This paragraph describes the Hanski connectivity on grid base maps where the maximum sum can be normalised to 1.

<u>LARCH-SCAN.exe:</u> This is the Windows GUI application for *WIN*

<u>LARCH_SCANc.exe</u>: This is the windows console version for *WIN*. The application needs an ini-file as an argument, like:

- "D:\...\LARCH_SCANc.exe" "D:\SCANTEST\0_scan_test_25_full_1_run.ini"

<u>Run ini-file:</u> The content ini file look like:

```
[scan_parameters]
population_map=D:\SCANTEST\0_scan_test_25_full_1.flt
scan_map=D:\SCANTEST\0_scan_test_25_full_1con3.flt
model_type=0
alpha=23.044094
percentage=95
normalise=1
max_density=1
extend_map=1
```

population_map: This file name refers to the existing floating point grid file. It contains the population quality or habitat suitability index for each cell. The maximum population/index value should be defined in [max_density]("max_density") when [normalise](normalise "normalise") is used.

scan_map: This file name refers to the resulting floating point grid file. It contains the SCAN values.

model_type: This defines the type of model in use:

- "model_type=0" defines the Hanski model, in this case the alpha and [percentage](percentage "percentage") are required as input variables.
- "model_type=1" defines the maximum distance model, in this case the [max_distance](max_distance "max_distance") is required as input variables.
- Alpha: Alpha in [connectivity_index](connectivity_index "connectivity index").

percentage: Percentage, otherwise the connectivity value continues to infinitely. Normally a percentage of 95% is used

max_distance: This is required as the maximum distance in model_type 1 normalise, Model selection:

- value 1 to normalise the SCAN Circle.
- value 0 to get the standard Hanski Connectivity.

max_density: If the habitat suitability value is not equal to 1 this value is required to normalise the SCAN. This is the maximum density for the "best" habitat type or the largest value in the map.

extend_map: If the habitat map is broken up into tiles the output should extent with the size of the connectivity circle. Set this value to 1, when working with tiles.

*4.3.4. LARCH-SCAN (Normalised) – *LINUX* - Step 02*

To execute:

./larch_scan population_map (without .flt suffix) scan_map (without .flt suffix) alpha_value percentage

Example for 10000m max distance:

./larch_scan input/class_20 output/class_20 0.29957 95.0

population_map: This file name refers to the existing floating point grid file. It contains the population quality or habitat suitability index for each cell. The maximum population/index value should be defined in [max_density]("max_density") when [normalise](normalise "normalise") is used.

scan_map: This file name refers to the resulting floating point grid file. It contains the SCAN values.

percentage: Percentage, otherwise the connectivity value continues to infinitely. A percentage of 95% is used by default.

alpha: The alpha value from the alpha table on the next page, that describes the examined distance. During the fully automatic processing on the DLR cluster these values were integrated into the program and always the following alpha values were calculated for each class: 5.991, 2.9957, 0.599 and 0.29957 at 95 percent.

#Hanski's connectivity index

Hanski's (1994 J Anim Ecol; see also Moilanen & Nieminen 2002 Ecology) connectivity index is computed as:

```
Ii = Σij exp(-αdij) × A × Hj
```

where:

dij is the distance between a given cell (i) and any other cell in the system (j), A is the (other) cell area, and Hj the Habitat Index value of the cell, α is the parameter defining the dispersal kernel, reflecting the probability that individuals reach a certain distance, and so it is needed to 'scale' the formula (because, e.g. 1 km isolation distance may be nothing for birds, but a serious isolation for snails).

Alpha's

In the table below, examples of Alpha values are listed. These values can easily be obtained by using the GUI and applying values to the maximum distance edit. The alpha changes accordingly. The Alpha values are calculated with the formula below:

```
alpha = (log(100)-log(100-pct))/distance(km)
```

or use the table below.

```
Max distance(m)  | Alpha(95%)      | Alpha(99%)
---------------- |---------------- |---------------
         50      |        59.91    |        92.10
         100     |        29.957   |        46.05
         250     |        11.982   |        18.42
         500     |         5.991   |         9.21
         1000    |         2.9957  |         4.60
         1500    |         1.9972  |         3.07
         5000    |         0.599   |         0.921
        10000    |         0.29957 |         0.460
        50000    |         0.0599  |         0.0921
```

This component is used up to approximately 10000 x 10000 cells on a standard desktop computer.

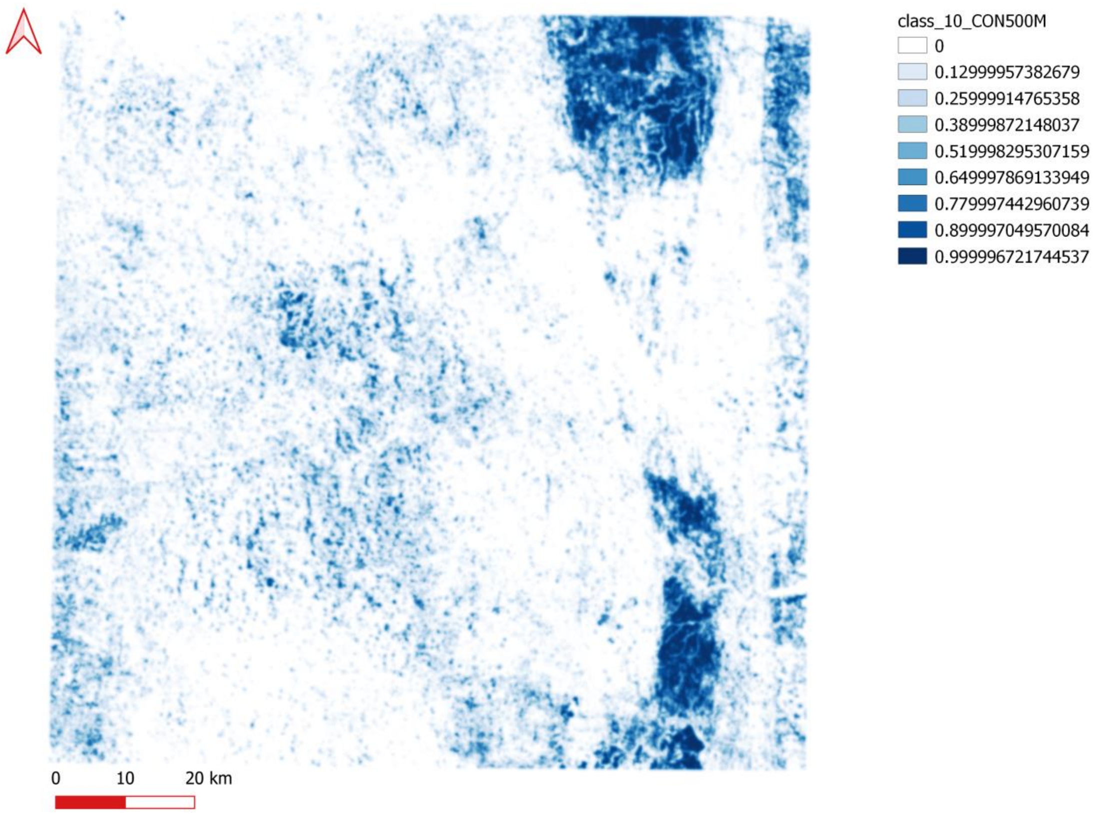

Figure 10: LARCH SCAN for FROM GLC10 JUT36 Sentinel 2 granule class 10 with alpha = 5.991 and percentage 95%.

*4.3.5. Select habitat and Calculate ClusterSize (ArcPy example)- *WIN* - Step 03*

The following example Python-script shows the algorithms for creating species specific output connectivity and cluster maps as it is used now in ArcMAP (ESRI). For the script code see chapter 9.3.

4.4. Performance gains over other algorithms

No other algorithms are described in this document. The LARCH-Hanski algorithm is a widely used and simple method (Opdam et al 2008, Pouwels et al, 2002). Its calculation is automated already in a standalone model, with applications at local, sub-national, national, continental and global scale. In the proposed stacked approach (see section 4.2) the main (time consuming) calculation process involves one step without manual intervention. After calculation of the stack of Hanski-fragmentation rasters just one intervention with ecoprofiles/species specific information is required to relate the calculated fragmentation rasters to biodiversity topics of choice.

Unlike other fragmentation indices, LARCH-Hanski is not sensitive for differences in raster resolution, and thus enables to calculate and compare fragmentation in both local as continental /global context (Pouwels et al, 2002).

5. Product

5.1. Product description

This section describes results of Fragmentation products retrieved from Proba-V land cover data (**Step01**) using the LARCH-SCAN algorithm. The LARCH-SCAN-Hanski metric can be calculated as a stack of spatial-temporal metrics based on multiple dispersal distances and (remotely derived) land-cover types (see Table 7). The spatial resolution vary by land cover product, with a target temporal resolution of one year.

Table 7: Stack of spatial cohesion products on basis of land cover types and dispersal distances as a result from Step02.

	Dispersal distance			
Land cover/habitat	***500***	***1km***	***5km***	***10km***
1	*SpatialCohesion_Type 1_ 500m*	*SpatialCohesion_Type1 _1km*	*SpatialCohesion_Type1_5 km*	*SpatialCohesion_Type1_1 0km*
2	*SpatialCohesion_Type 2_ 500m*	*SpatialCohesion_Type2 _1km*	*SpatialCohesion_Type2_5 km*	*SpatialCohesion_Type2_1 0km*
3	*SpatialCohesion_Type 3_ 500m*	*SpatialCohesion_Type3 _1km*	*SpatialCohesion_Type3_5 km*	*SpatialCohesion_Type3_1 0km*
4	*etc*	*Etc*	*etc*	*etc*
....	*.....*	*.....*	*.....*	*....*

Step02 result: In total 14 land cover types x 4 distances == 56 spatial cohesion products

This approach gives us the maximum flexibility, not only the calculate the spatial cohesion per habitat or land cover type, but also to combine the individual results for a specific flagship species or species groups. Since species can require a combination of land cover types in their habitat, we can sum-up the above mentioned individual spatial cohesion maps to a maximum of one.

Typical output showing maps of spatial cohesion:

- LARCH Hanski metric ranging from 0-1 for each class in the map, for each selected distance
- LARCH Hanski metric ranging from 0-1 for a (weighted) selection of classes x distance related to the selected ecoprofile

Step03 (ecoprofile Black Rhino) result: in

Summation of 3 land cover types, types x 1 distances == 1 connectivity product (see Figure 11, left)

Step04 (optional): ecoprofile Black Rhino clusters

- clusters related to the selected ecoprofile
- Size in ha (when using a projected coordinate system) of the clusters (here applied), see Figure 11, right

Count nr of cells (when using a geographic coordinate system) of the clusters (here not applied)

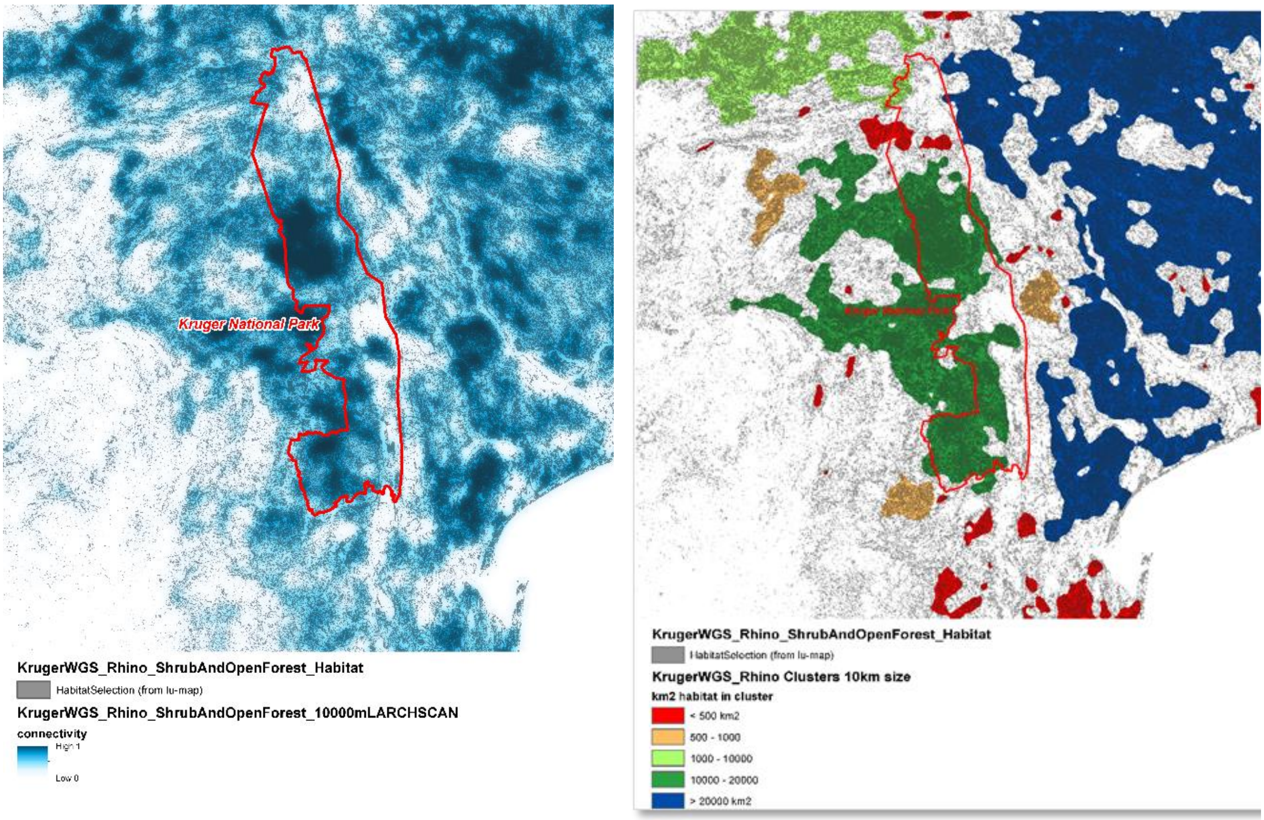

Figure 11: (left) Example of typical final output map (example ecoprofile Black Rhino) of spatial cohesion for Tropical/ sub-tropical, shrubland habitat in Kruger National Park, South Africa based on a selection of habitat classes (shrub and open forest), for a specific cohesion distance (10000m). (right) Clusters represent very good connected habitat with a threshold > 0.5.

5.2. Validation

Assessment of the implemented algorithm itself is possible by comparing the EBV fragmentation product for the ecoprofile "Black Rhino" with the current and historical distribution of the Black Rhino (IUCN African Rhino Specialist Group, 2018, Rookmaker & Antoine, 2012), see also Figure 12.

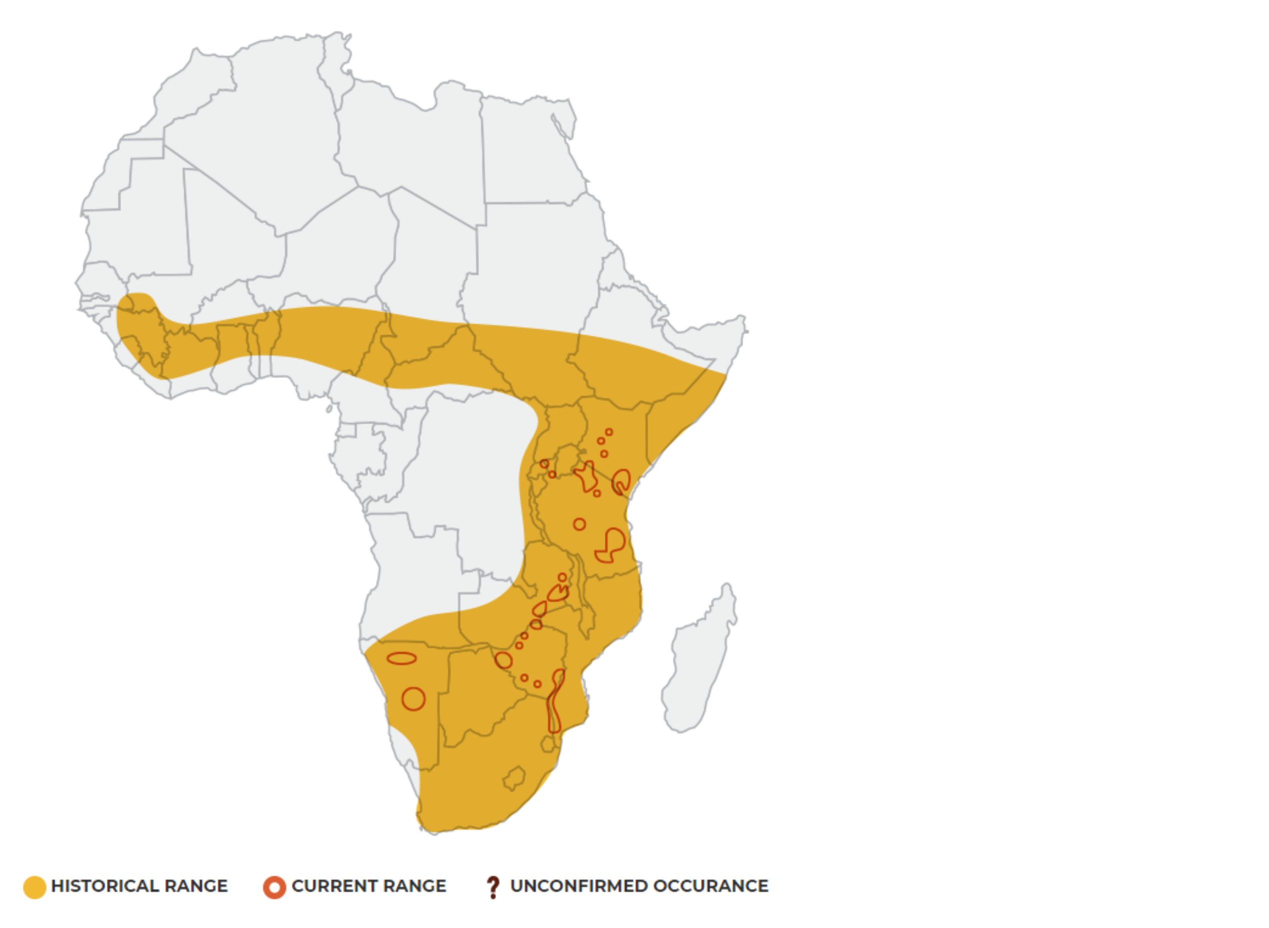

Figure 12: Historical and current distribution of the black rhinoceros (https://rhinos.org/species/black-rhino/, IUCN African Rhino Specialist Group, 2016, Rookmaker & Antoine, 2012).

6. Practical considerations for implementation

6.1. Memory requirements

Due to the high resolution, only a small amount of 4 or 8 GB RAM is required for high alpha values. The smaller the alpha value becomes, the higher the memory consumption becomes up to 64 or 128 GB of RAM.

6.2. System requirements

The system requires a fast processor with high single threading performance for high alpha values. For distances like 500 or 1000 meters a normal notebook with a recent Intel Core i5 computer is sufficient. The system should be able to cope with longer running times without problems. Optimal is therefore a server or a specially installed processing system. Only Windows (*WIN*) or Windows and Linux (*LINUX*) can be used as operating system. The current compiler tools are required, which is able to build standard programs with C++17 (*LINUX*) or Borland compiler (*WIN*).

6.3. Error handling

During prototyping, no explicit attention was paid to error handling. If input values are incorrect, the program simply terminates and displays error messages in rare cases.

6.4. External databases

No external databases are necessary.

6.5. Manual interaction

No manual intervention is necessary during processing. At the beginning, the appropriate configuration files must be stored and transferred to the program (*WIN* with .ini file approach) or the call of the program must be controlled with shell scripts (*LINUX* approach).

6.6. Algorithm validation

NA

6.7. Numerical computing considerations

During processing it has been shown that for small alpha values the calculation time is dramatically increased and therefore it would make sense to use tiling, multiprocessing, map and reduce or support by graphics processors (CUDA) in the next step. Alternatively it would make sense to use a machine learning approach.

See provided source code and workflows.

Concerning algorithm validation the output produced on basis of the ATBD can be validated with earlier calculations done with standalone version of LARCH SCAN for Black Rhino in test site Kruger Park.

7. Upscaling results

7.1. Introduction

Spatial cohesion maps and habitat fragmentation maps were retrieved for two pre-defined areas of interests, i.e. Finland and Senegal. In order to calculate these maps, the Copernicus Global Land Service 100 m collection 3 land cover maps for 2019 (Buchhorn et al. 2020) produced by VITO was used as input land cover layer (Figure 13).

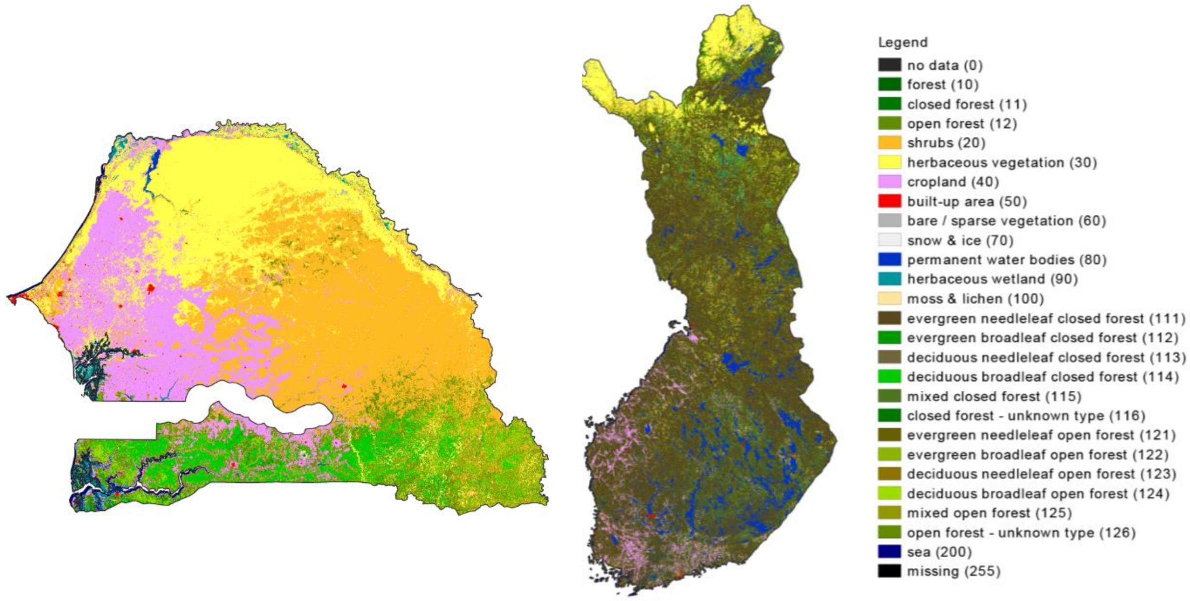

Figure 13: 2019 Copernicus Global Land Service Collection 3 for 2019 at 100 m resolution available from https://land.copernicus.eu/global/products/lc

Spatial cohesion was calculated for all land cover classes at the following dispersal distances: 500 m, 1000 m, 5000 m and 10 000 m. Next connectivity and cluster maps were determined for 2 different ecoprofiles: Chimpanzee (Senegal) and Arctic Fox (Finland)

7.2. Results

Figure 14 and Figure 15 illustrate the resulting spatial cohesion maps for all different land cover classes at a dispersal distance of 10 000 m over Finland and Senegal, respectively. Figure 16 on the other hand shows the calculated spatial cohesion for one particular class ("shrubs") over Senegal, clearly showing the effect of an increasing dispersal distance on the obtained cohesion maps.

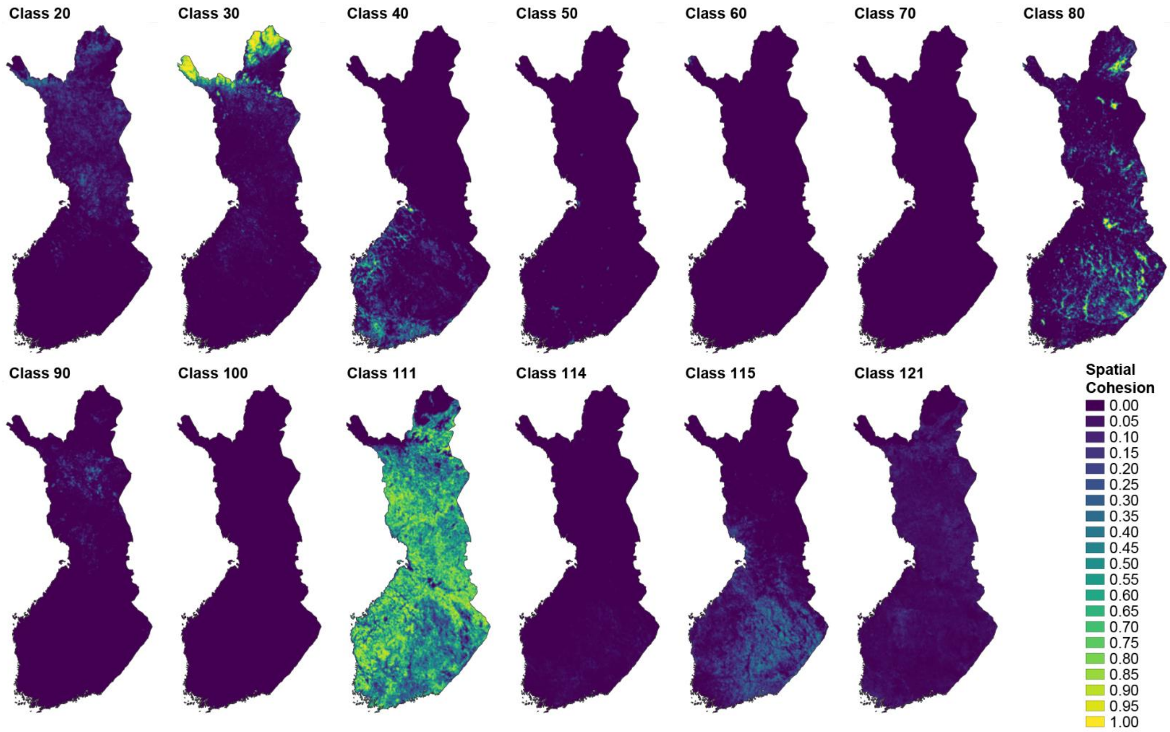

Figure 14: Spatial cohesion maps for all different land cover classes over Finland (see Figure 13 for the corresponding class to the class number) calculated with a dispersal distance of 10 000m.

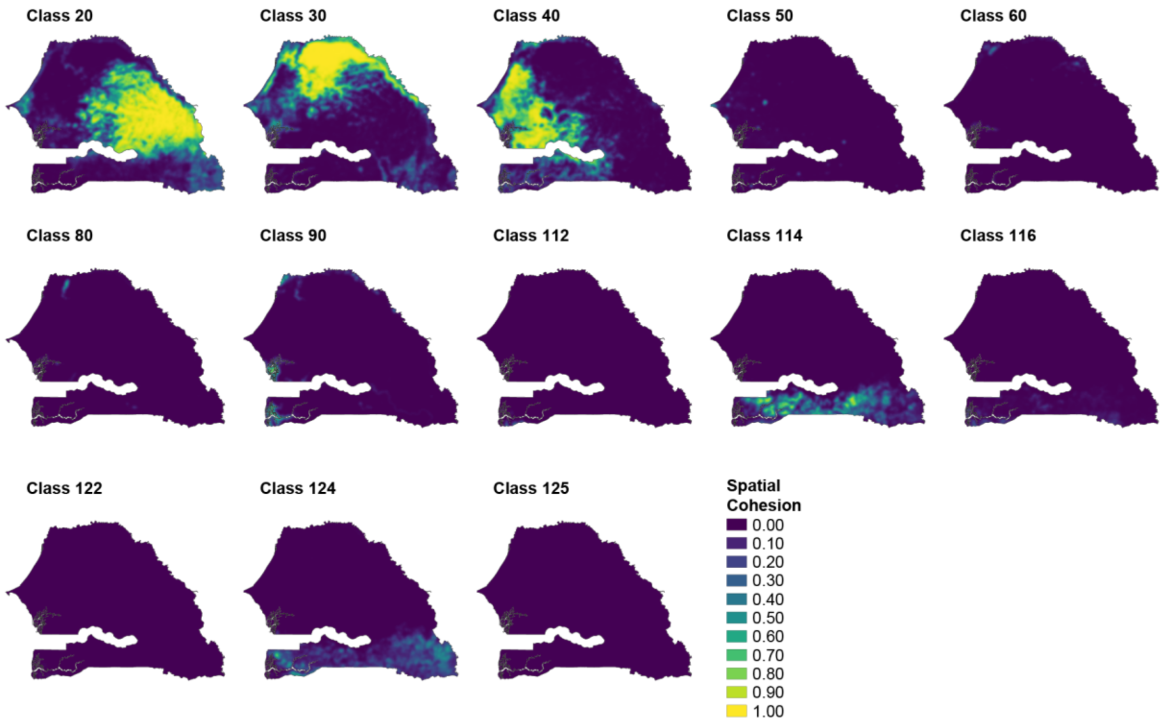

Figure 15: Spatial cohesion maps for all different land cover classes over Senegal(see Figure 13 for the corresponding class to the class number) calculated with a dispersal distance of 10 000m.

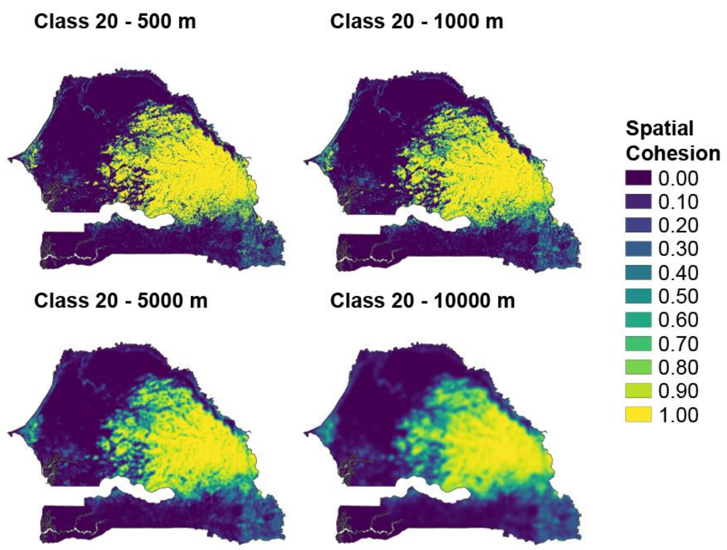

Figure 16: Spatial cohesion maps at different dispersal distances for the class "shrubs".

Based on those spatial cohesion maps, habitat clusters were calculated for the Chimpanzee (Senegal) and Arctic Fox (Finland) ecoprofiles. Table 8 lists the different parameters that were used in this calculation. **Fehler! Verweisquelle konnte nicht gefunden werden.** finally, illustrates these layers showing the resulting potential habitat areas for the 2 species.

Table 8: Ecoprofiles and parameters used for Chimpanzee and Arctic Fox

Parameter	**Chimpanzee**	**Arctic Fox**
Considered classes	All deciduous forest classes (i.e.class 114 and 124)	Herbaceous vegetation and lichen & mosses (i.e. class 30 and 100)
Considered dispersal distances	500 m, 1000 m and 5000 m	500 m, 1000 m and 5000 m
Minimal spatial cohesion threshold	0.1	0.1
Min cluster size/home range	6 km²	15 km²

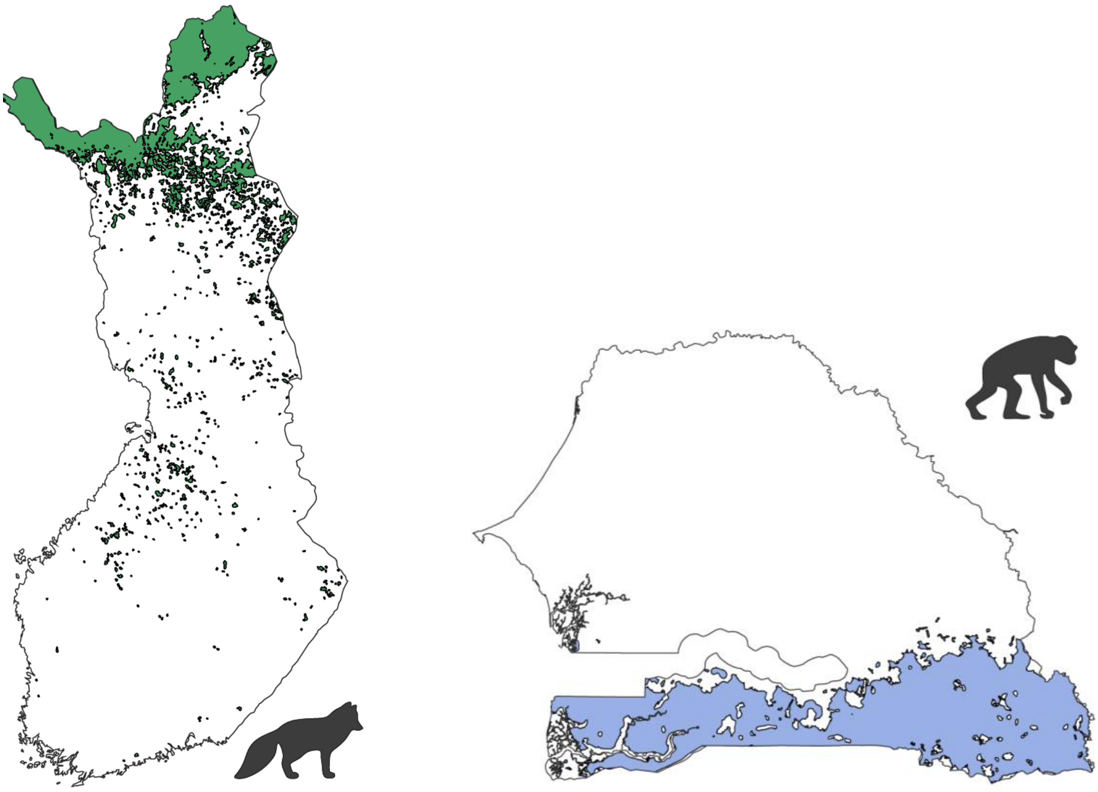

Figure 17: Clusters representing connected habitat with a threshold > 0.1 for Arctic Fox (green) and Chimpanzee (blue)

8. References

Beissinger, S.R. & McCullough, D.R. (2002) Population viability analysis. University of Chicago Press.

Buchhorn, M. ; Smets, B. ; Bertels, L. ; De Roo, B. ; Lesiv, M. ; Tsendbazar, N. - E. ; Herold, M. ; Fritz, S. Copernicus Global Land Service: Land Cover 100m: collection 3: epoch 2019: Globe 2020.

EEA/FOEN (2011) Landscape fragmentation in Europe. Copenhagen.

Emslie, R. 2012. Diceros bicornis. The IUCN Red List of Threatened Species 2012: e.T6557A16980917. http://dx.doi.org/10.2305/IUCN.UK.2012.RLTS.T6557A16980917.en.

Fahrig, L. (2003) Effects of habitat fragmentation on biodiversity. Annual review of ecology, evolution, and systematics, 487-515.

Forman, R.T. & Alexander, L.E. (1998) Roads and their major ecological effects. Annual Review of Ecology and Systematics, 207-C202.

Groot Bruinderink, G.W.T.A., T. van der Sluis, D.R. Lammertsma, P. Opdam. and R. Pouwels. 2003. The assessment of a tentative, coherent ecological network for large mammals in Northwest Europe. Biological Conservation 17(2): 549-557.

Groot Bruinderink, G.W.T.A., T. van der Sluis, D.R. Lammertsma, P. Opdam. and R. Pouwels. 2003. The assessment of a tentative, coherent ecological network for large mammals in Northwest Europe. Biological Conservation 17(2): 549-557.

Hansen, M. C., P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, et al. 2013. High-resolution global maps of 21st century forest cover change. Science 342, 850–853.

Hanski, I. (2011) Habitat loss, the dynamics of biodiversity, and a perspective on conservation. AMBIO: A Journal of the Human Environment, 40, 248-255.

Hanski, I. 1994. A practical model of metapopulation dynamics. Journal of Animal Ecology 63: 151-162.

Hodgson, J.A., Moilanen, A., Wintle, B.A. & Thomas, C.D. (2011) Habitat area, quality and connectivity: striking the balance for efficient conservation. Journal of Applied Ecology, 48, 148-152.

IEEP & Alterra (2010) Reflecting environmental land use needs into EU policy: preserving and enhancing the environmental benefits of “land services”: soil sealing, biodiversity corridors, intensification / marginalisation of land use and permanent grassland. Final report to the European Commission, DG Environment on Contract ENVB1/ETU/2008/0030. Institute for European Environmental Policy / Alterra Wageningen UR.

IUCN African Rhino Specialist Group, 2018. https://rhinos.org/research-publications/iucn-african-rhino-specialist-group/

Jaeger, J.A.G. (2000) Landscape division, splitting index, and effective mesh size: new measures of landscape fragmentation. Landscape Ecology, 15, 115-130.

Lent P.C. & B. Fike, 2003.Home ranges, movements and spatial relationships in an expanding population of black rhinoceros in the Great Fish River Reserve, South Africa. South African Journal of Wildlife Research Vol. 33, No. 2, October 2003

Linklater W.L. & I.R. Hutcheson, 2010. Black rhinoceros are slow to colonize a harvested neighbour's range. South African Journal of Wiidiife Research 40(1): 58-63 (Aprii 2010)

Margules, C.R. & Pressey, R.L. (2000) Systematic conservation planning. Nature, 405, 243-253.

Neel, M.C., McGarigal, K. & Cushman, S.A. (2004) Behavior of class-level landscape metrics across gradients of class aggregation and area. Landscape Ecology, 19, 435-455.

Nilsson, C., Reidy, C.A., Dynesius, M. & Revenga, C. (2005) Fragmentation and flow regulation of the world's large river systems. Science, 308, 405-408.

Opdam, P. (1991) Metapopulation theory and habitat fragmentation - a review of holarctic breeding bird studies. Landscape Ecology, 5, 93-106.

Opdam, P., Pouwels, R., van Rooij, S., Steingrover, E. & Vos, C.C. (2008) Setting Biodiversity Targets in Participatory Regional Planning: Introducing Ecoprofiles. Ecology and Society, 13.

Opdam, P., Verboom, J. & Pouwels, R. (2003) Landscape cohesion: an index for the conservation potential of landscapes for biodiversity. Landscape Ecology, 18, 113-126.

Ouborg, N.J. (1993) Isolation, population size and extinction: the classical and metapopulation approaches applied to vascular plants along the Dutch Rhine-system. Oikos, 298-308.

Pereira, H. M., S. Ferrier, M. Walters, G. N. Geller, R. H. G. Jongman, R. Scholes, 2013. Essential biodiversity variables. Science 339, 277–278.

Pereira, H.M., Leadley, P.W., Proença, V., Alkemade, R., Scharlemann, J.P., Fernandez-Manjarrés, J.F., Araújo, M.B., Balvanera, P., Biggs, R. & Cheung, W.W. (2010) Scenarios for global biodiversity in the 21st century. Science, 330, 1496-1501.

Plotz RD, Grecian WJ, Kerley GIH, Linklater WL (2016) Standardising Home Range Studies for Improved Management of the Critically Endangered Black Rhinoceros. PLoS ONE 11(3): e0150571. doi:10.1371/journal.pone.0150571

Pouwels, R., R. Jochem, M.J.S.M. Reijnen, S.R. Hensen en J.G.M. van der Greft. 2002. LARCH voor ruimtelijk ecologische beoordelingen van landschappen. Alterra-rapport 492. Alterra, Research Instituut voor de Groene Ruimte, Wageningen. [In Dutch]

Riitters, K.H., Oneill, R.V., Hunsaker, C.T., Wickham, J.D., Yankee, D.H., Timmins, S.P., Jones, K.B. & Jackson, B.L. (1995) A factor-analysis of landscape pattern and structure metrics. Landscape Ecology, 10, 23-39.

Rookmaaker, K. & P.O. Antoine 2012. New maps representing the historical and recent distribution of the African species of rhinoceros: Diceros bicornis, Ceratotherium simum and Ceratotherium cottoni. Pachyderm 52, 91-96

Rybicki, J. & Hanski, I. (2013) Species–area relationships and extinctions caused by habitat loss and fragmentation. Ecology Letters.

Schipper, J., Chanson, J.S., Chiozza, F., Cox, N.A., Hoffmann, M., Katariya, V., Lamoreux, J., Rodrigues, A.S.L., Stuart, S.N., Temple, H.J., Baillie, J., Boitani, L., Lacher, T.E., Mittermeier, R.A., Smith, A.T., Absolon, D., Aguiar, J.M., Amori, G., Bakkour, N., Baldi, R., Berridge, R.J., Bielby, J., Black, P.A., Blanc, J.J., Brooks, T.M., Burton, J.A., Butynski, T.M., Catullo, G., Chapman, R., Cokeliss, Z., Collen, B., Conroy, J., Cooke, J.G., da Fonseca, G.A.B., Derocher, A.E., Dublin, H.T., Duckworth, J.W., Emmons, L., Emslie, R.H., Festa-Bianchet, M., Foster, M., Foster, S., Garshelis, D.L., Gates, C., Gimenez-Dixon, M., Gonzalez, S., Gonzalez-Maya, J.F., Good, T.C., Hammerson, G., Hammond, P.S., Happold, D., Happold, M., Hare, J., Harris, R.B., Hawkins, C.E., Haywood, M., Heaney, L.R., Hedges, S., Helgen, K.M., Hilton-Taylor, C., Hussain, S.A., Ishii, N., Jefferson, T.A., Jenkins, R.K.B., Johnston, C.H., Keith, M., Kingdon, J., Knox, D.H., Kovacs, K.M., Langhammer, P., Leus, K., Lewison, R., Lichtenstein, G., Lowry, L.F., Macavoy, Z., Mace, G.M., Mallon, D.P., Masi, M., McKnight, M.W., Medellin, R.A., Medici, P., Mills, G., Moehlman, P.D., Molur, S., Mora, A., Nowell, K., Oates, J.F., Olech, W., Oliver, W.R.L., Oprea, M., Patterson, B.D., Perrin, W.F., Polidoro, B.A., Pollock, C., Powel, A., Protas, Y., Racey, P., Ragle, J., Ramani, P., Rathbun, G., Reeves, R.R., Reilly, S.B., Reynolds, J.E., Rondinini, C., Rosell-Ambal, R.G., Rulli, M., Rylands, A.B., Savini, S., Schank, C.J., Sechrest, W., Self-Sullivan, C., Shoemaker, A., Sillero-Zubiri, C., De Silva, N., Smith, D.E., Srinivasulu, C., Stephenson, P.J., van Strien, N., Talukdar, B.K., Taylor, B.L., Timmins, R., Tirira, D.G., Tognelli, M.F., Tsytsulina, K., Veiga, L.M., Vie, J.C., Williamson, E.A., Wyatt, S.A., Xie, Y. & Young, B.E. (2008) The status of the world's land and marine mammals: Diversity, threat, and knowledge. Science, 322, 225-230.

Siefke, A. 1984. Zur Dismigration der Vögel als popularem Phänomen I. Ein heuristisches Modell der Ansiedlerstreuung. Zool. Jb. Syst. 111: 307-319.

Snyder J.P. 1987. Map Projections, A Working Manual U.S. GEOLOGICAL SURVEY PROFESSIONAL PAPER 1395 Available at https://pubs.usgs.gov/pp/1395/report.pdf

Taylor, B.D. & Goldingay, R.L. (2010) Roads and wildlife: impacts, mitigation and implications for wildlife management in Australia. Wildlife Research, 37, 320-331.

IUCN African Rhino Specialist Group, 2016 https://rhinos.org/research-publications/iucn-african-rhino-specialist-group/

Thomas, C.D. & Hanski, I. (1997) Butterfly metapopulations. Metapopulation Biology: Ecology, Genetics, and Evolution (eds I. Hanski & M.E. Gilpin), pp. 359-386. Academic Press, San Diego.

Van Der Grift, E. & Pouwels, R. (2006) Restoring habitat connectivity across transport corridors: Identifying high-priority locations for de-fragmentation with the use of an expert-based model. Ecology of Transportation: Managing Mobility for the Environment, 10, 205-231.

Verboom, J. and R. Pouwels. 2004. Ecological functioning of ecological networks: a species perspective. In: Jongman en Pungetti (eds.). Ecological Networks and Greenways: concept, design, implementation. Cambridge University Press, Cambridge.

Verboom, J., R. Foppen , J.P. Chardon, P.F.M. Opdam en P.C. Luttikhuizen. 2001. Introducing the key patch approach for habitat networks with persistent populations: an example for marshland birds. Biological Conservation. Vol 100 (1). pp. 89-100

Villard, M.A., Trzcinski, M.K. & Merriam, G. (1999) Fragmentation effects on forest birds: relative influence of woodland cover and configuration on landscape occupancy. Conservation Biology, 13, 774-783.

Vos, C.C., Opdam, P., Steingröver, E.G. & Reijnen, R. (2007) Transferring ecological knowledge to landscape planning: a design method for robust corridors. Integrating landscape ecology into nature resource management (eds J. Liu & W.W. Tayler), pp. 227-245. Cambridge University Press, Cambridge.

Vos, C.C., Verboom, J., Opdam, P.F.M. & Ter Braak, C.J.F. (2001) Toward ecologically scaled landscape indices. American Naturalist, 157, 24-41.

9. Appendix

9.1. Split classes Habitat map *WIN* - Step 01

The following example Python-script shows the algorithm for creating input splitted habitat maps as it is used in ArcMAP (ESRI) by WEnR.

```
# -*- coding: utf-8 -*-
# ---------------------------------------------------------------------------
# SplitHabitat.py
# Created on: 2018-11-19 15:14:21.00000
# Usage: SplitHabitat <inHabitats_LandcoverClasses> <NameArea> <OutputFolder>
# Description:
# Create_FLTs
# ---------------------------------------------------------------------------

# Import arcpy module
import arcpy

# Load required toolboxes
arcpy.ImportToolbox("Model Functions")

# Script arguments
inHabitats_LandcoverClasses = arcpy.GetParameterAsText(0)
if inHabitats_LandcoverClasses == '#' or not inHabitats_LandcoverClasses:
 inHabitats_LandcoverClasses = "[Habitat/LandCover-Map].tif" # provide a default value if unspecified

NameArea = arcpy.GetParameterAsText(1)
if NameArea == '#' or not NameArea:
 NameArea = "[NAME]" # provide a default value if unspecified

OutputFolder = arcpy.GetParameterAsText(2)
if OutputFolder == '#' or not OutputFolder:
 OutputFolder = "[FOLDER]" # provide a default value if unspecified

# Local variables:
HabClass = inHabitats_LandcoverClasses
Extract_HabClass = "%OutputFolder%\\Extract_HabClass.tif"
Reclass_Habitat = "%OutputFolder%\\Reclass_Habitat.tif"
v_NameArea_HabClass_flt = "%OutputFolder%\\%NameArea%_%HabClass%.flt"
```

```
# Process: Iterate Field Values

arcpy.IterateFieldValues_mb(inHabitats_LandcoverClasses, "Value", "Long", "true",
"false", "0")


# Process: Extract by Attributes

arcpy.gp.ExtractByAttributes_sa(inHabitats_LandcoverClasses, "\"Value\" = %HabClass%",
Extract_HabClass)


# Process: Reclassify

arcpy.gp.Reclassify_sa(Extract_HabClass, "Value", "-9999999999 10000000000 1",
Reclass_Habitat, "NODATA")


# Process: Raster to Float

arcpy.RasterToFloat_conversion(Reclass_Habitat, v_NameArea_HabClass_flt)
```

9.2. Split classes Habitat map *LINUX* - Step 01

The following Python-script contains the code for creating splitted habitat maps as it is used and required for the subsequent LARCH-SCAN *LINUX* step 02.

```
__status__ = "Prototype"


from src.gdal_functions import *
import argparse
from osgeo.gdalnumeric import *
from osgeo import gdal
import numpy as np
import logging
import glob
import sys
import os



class Raster:
    def __init__(self):
        self.raster_file = None
        self.output_path = None
        self.granule = None
        self.x_size = None
        self.y_size = None
        self.geo_transform = None
        self.projection = None
        self.raster_file_band = None
```

```
def create_log(log_object):
    """
    Create a log object to avoid print statements, ability of debug statements and log
file creation

    :param log_object: Current name of the function
    :return: log object with parameters regarding log level and with formatting options
    """

    # Set log level to lowest level for correct debug file logging, kind of global
logger option
    log_object.setLevel(level=logging.INFO)

    # Create console handler with a higher log level
    stream_handler = logging.StreamHandler()
    stream_handler.setLevel(level=logging.DEBUG)

    # Create formatter and add it to the handlers
    formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s -
%(message)s')

    # Set formatting
    stream_handler.setFormatter(formatter)

    # Add the handlers to logger
    log_object.addHandler(stream_handler)

    return log_object


def split(llc_file, args, log_object):

    # Create Raster Class
    raster = Raster()

    # this allows GDAL to throw Python Exceptions
    gdal.UseExceptions()

    # Append current working directory
    sys.path.append(os.getcwd())
```

```
    log_object.debug("Append current working directory to path: " + str(os.getcwd()))

    # File IO settings
    raster.input_raster = llc_file
    log_object.debug('Input raster is: ' + str(raster.input_raster))

    raster.granule = os.path.splitext(str(os.path.basename(llc_file)))[0]
    log_object.debug('Granule is: ' + str(raster.granule))

    raster.output_path = os.path.join(args.result + raster.granule)
    log_object.debug('Output path is: ' + str(raster.output_path))

    # Create folder for output files
    if not os.path.exists(raster.output_path):
        os.mkdir(raster.output_path)
        log_object.info("Directory " + str(raster.output_path) + " created ")
    else:
        log_object.debug("Directory " + str(raster.output_path) + " already exists -
Skipping creation of folder")

    # Open the raster layer
    raster.raster_file = open_raster_file(input_raster=raster.input_raster)
    log_object.debug('Input raster successfully imported:
{raster}'.format(raster=raster.raster_file))

    # Get relevant properties
    raster.x_size = get_x_size(raster_file=raster.raster_file)
    log_object.debug('Raster X Size: {x}'.format(x=raster.x_size))
    raster.y_size = get_y_size(raster_file=raster.raster_file)
    log_object.debug('Raster Y Size: {y}'.format(y=raster.y_size))
    raster.geo_transform = get_geo_transform(raster_file=raster.raster_file)
    log_object.debug('Raster Geo Transform: {trans}'.format(trans=raster.geo_transform))
    raster.projection = get_projection(raster_file=raster.raster_file)
    log_object.debug('Raster Projection: {proj}'.format(proj=raster.projection))

    # Load Bands
    raster.raster_file_band = get_raster_band(raster_file=raster.raster_file)
    log_object.debug('Band successfully loaded
{band}'.format(band=raster.raster_file_band))

    # Get unique raster values from band and scene
    unique_raster_values = np.unique(gdal.Band.ReadAsArray(raster.raster_file_band))
```

```
    log_object.debug('Got all unique values from raster:
{unique}'.format(unique=unique_raster_values))

    length = len(unique_raster_values)

    log_object.debug('Amount of unique values from raster:
{amount}'.format(amount=length))


    # Debug information

    log_object.debug('[ X Size ] = {x}'.format(x=raster.x_size))

    log_object.debug('[ Y Size ] = {y}'.format(y=raster.y_size))

    log_object.debug('[ GeoTransform ] = {trans}'.format(trans=raster.geo_transform))

    log_object.debug('[ Projection ] =
{projection}'.format(projection=raster.projection))

    log_object.debug("[ NO DATA VALUE ] =
{no_data}".format(no_data=raster.raster_file_band.GetNoDataValue()))

    log_object.debug("[ MIN ] = {min}".format(min=raster.raster_file_band.GetMinimum()))

    log_object.debug("[ MAX ] = {max}".format(max=raster.raster_file_band.GetMaximum()))

    log_object.debug("[ SCALE ] =
{scale}".format(scale=raster.raster_file_band.GetScale()))

    log_object.debug("[ UNIT TYPE ] =
{unit}".format(unit=raster.raster_file_band.GetUnitType()))


    # Convert to array

    raster.raster_array = read_gdal_object_as_array(gdal_object=raster.raster_file_band)

    log_object.debug('Reading of GDAL array successful')


    # Value of -9999 did not work out, since there is a conversion to a another value
like 55725

    # Value of -9999 is not able to be set as NoDataValue (no error, but some hidden
magic happens here)

    nodata_value = 0

    log_object.debug('NoData value was set to: {nodata}'.format(nodata=nodata_value))


    # Save classes as individual files

    for i in range(length):


        if unique_raster_values[i] == -9999.0:

            log_object.debug("Skipping NoData value calculation -9999.0")

            continue


        if unique_raster_values[i] == 0:

            log_object.debug("Skipping Class 0 value calculation")

            continue


        log_object.debug('Writing class number
{number}'.format(number=int(unique_raster_values[i])))
```

```
        output_file = np.full_like(raster.raster_array, nodata_value)
        log_object.debug('Created empty raster with NoData values
({nodata})'.format(nodata=nodata_value))

        output_file[raster.raster_array == unique_raster_values[i]] = 1
        log_object.debug('Set empty raster to 1 if class condition was met')

        # Write the out file, GTiff (.tif) or ENVI (.bin) or EHdr (.flt)
        driver = gdal.GetDriverByName("EHdr")
        output_name = os.path.join(raster.output_path, str("class_") +
str(int(unique_raster_values[i])) + str(".flt"))
        log_object.debug('Output file name is {file}'.format(file=output_name))

        # Options: bands=1, eType=gdal.GDT_Byte, options=['COMPRESS=LZW'])
        gdal_dataset = driver.Create(output_name,
                                     xsize=raster.x_size,
                                     ysize=raster.y_size,
                                     bands=1,
                                     eType=gdal.GDT_Float32)

        out_band = get_raster_band(gdal_dataset)
        out_band.SetNoDataValue(nodata_value)
        out_band.WriteArray(output_file)

        # flush data to disk, set the NoData value and calculate stats
        out_band.FlushCache()

        # Georeference the image and set the projection
        # Important do not delete this statement!
        gdal_dataset.SetGeoTransform(raster.geo_transform)
        gdal_dataset.SetProjection(raster.projection)

def main():
    # Create Parser and Log object
    parser = argparse.ArgumentParser()
    log_object = create_log(logging.getLogger(__name__))

    # Add parser arguments
    parser.add_argument("llc", help="full path to local landcover TIF files")
    parser.add_argument("result", help="full path to result output folder")
    parser.add_argument("--debug", help="show debug information", action="store_true")
```

```
    # Create parser objects
    args = parser.parse_args()

    # Check and change for and to debug mode
    if args.debug:
        log_object.setLevel(level=logging.DEBUG)
        log_object.info("Set log level to DEBUG")

    # Platform information
    log_object.debug("General platform information")
    log_object.debug("Version 
{major_version}.{minor_version}".format(major_version=sys.version_info.major,

minor_version=sys.version_info.minor))
    log_object.debug("Platform {platform}".format(platform=sys.platform))
    log_object.debug("Current working directory: " + str(os.getcwd()))

    # Append current working directory
    sys.path.append(os.getcwd())
    log_object.debug("Append current working directory to path: " + str(os.getcwd()))

    # Are the paths correct?
    log_object.debug("Path to local landcover file folder is 
{llc}".format(llc=args.llc))
    log_object.debug("Path to result output folder is 
{result}".format(result=args.result))

    llc_files = glob.glob(args.llc + "*.tif", recursive=True)
    log_object.debug("Input file list is " + str(llc_files))

    for llc_file in llc_files:
        log_object.info("Processing file: " + str(llc_file))
        split(llc_file=llc_file, args=args, log_object=log_object)

if __name__ == "__main__":
    main()
```

9.3. Select habitat and Calculate ClusterSize (ArcPy example)- *WIN* - Step 03

The following example Python-script shows the algorithms for creating species specific output connectivity and cluster maps as it is used now in ArcMAP (ESRI). For the script code see chapter

```
# ---------------------------------------------------------------------------
```

```
# Calc_habsize.py
# Created on: 2018-09-20 11:11:55.00000
# # Usage: Calc_habsize <Select_Habitat_from_map> <Select_Habitatclasses>
  <Select_Connectivity_Files> <OutputFolder> <NameArea> <HabitatName>
  <Connectivity_Distance> <LARCHScan_Threshold> <Spatial_Reference_for_Raster> <CellSize_m>
  <Mask>
# ---------------------------------------------------------------------------


# Import arcpy module
import arcpy


# Script arguments
Select_Habitat_from_map = arcpy.GetParameterAsText(0)
if Select_Habitat_from_map == '#' or not Select_Habitat_from_map:
 Select_Habitat_from_map = "OCS_2017_Cesbio_JRC_20m_2.tif" # provide a default value if
  unspecified


Select_Habitatclasses = arcpy.GetParameterAsText(1)
if Select_Habitatclasses == '#' or not Select_Habitatclasses:
 Select_Habitatclasses = "[classes]" # provide a default value if unspecified


Select_Connectivity_Files = arcpy.GetParameterAsText(2)
if Select_Connectivity_Files == '#' or not Select_Connectivity_Files:
 Select_Connectivity_Files = "[files].flt VALUE 1" # provide a default value if unspecified


OutputFolder = arcpy.GetParameterAsText(3)
if OutputFolder == '#' or not OutputFolder:
 OutputFolder = "[folder]" # provide a default value if unspecified


NameArea = arcpy.GetParameterAsText(4)
if NameArea == '#' or not NameArea:
 NameArea = "[NameArea]" # provide a default value if unspecified


HabitatName = arcpy.GetParameterAsText(5)
if HabitatName == '#' or not HabitatName:
 HabitatName = "[HabName]" # provide a default value if unspecified


Connectivity_Distance = arcpy.GetParameterAsText(6)
if Connectivity_Distance == '#' or not Connectivity_Distance:
 Connectivity_Distance = ""[ConDistName]"" # provide a default value if unspecified


LARCHScan_Threshold = arcpy.GetParameterAsText(7)
if LARCHScan_Threshold == '#' or not LARCHScan_Threshold:
```

```
 LARCHScan_Threshold = ""[Value 0-1]"" # provide a default value if unspecified

Spatial_Reference_for_Raster = arcpy.GetParameterAsText(8)
if Spatial_Reference_for_Raster == '#' or not Spatial_Reference_for_Raster:
 Spatial_Reference_for_Raster = "[CoordinateSystem]" # provide a default value if
   unspecified

CellSize_m = arcpy.GetParameterAsText(9)
if CellSize_m == '#' or not CellSize_m:
 CellSize_m = "[Cellsize (m)]" # provide a default value if unspecified

Mask = arcpy.GetParameterAsText(10)
if Mask == '#' or not Mask:
 Mask = "[MaksRaster]" # provide a default value if unspecified

# Local variables:
Mosaic = Mask
v_NameArea_HabitatName_Connectivity_Distance_LARCHSCAN_RAW_tif =
   "%OutputFolder%\\%NameArea%_%HabitatName%_%Connectivity Distance%LARCHSCAN_RAW.tif"
tmp_0_tif = "%OutputFolder%\\tmp_0.tif"
IntMosaic = "%OutputFolder%\\%NameArea%_%HabitatName%_%Connectivity Distance%.tif"
Input_true_raster_or_constant_value = "1"
tmp_nodata_tif = "%OutputFolder%\\tmp_nodata.tif"
Threshold = "\"VALUE\" > %LARCHScan Threshold%"
tmp_clust1_tif = "%OutputFolder%\\tmp_clust1.tif"
v_NameArea_HabitatName_HabitatClusters_tif =
   "%OutputFolder%\\%NameArea%_%HabitatName%_HabitatClusters.tif"
tmp_clustjoin = v_NameArea_HabitatName_HabitatClusters_tif
clust1 = v_NameArea_HabitatName_HabitatClusters_tif
v_NameArea_HabitatName_HabitatQual_tif =
   "%OutputFolder%\\%NameArea%_%HabitatName%_HabitatQual.tif"
HabQual_sumtable = "%OutputFolder%\\HabQual_sumtable"
clust2 = HabQual_sumtable
clust3 = clust2
v_NameArea_HabitatName_Connectivity_Distance_HabSizeHa_tif =
   "%OutputFolder%\\%NameArea%_%HabitatName%_%Connectivity Distance%_HabSizeHa.tif"

# Process: Weighted Sum
tempEnvironment0 = arcpy.env.outputCoordinateSystem
arcpy.env.outputCoordinateSystem = Spatial Reference for Raster
tempEnvironment1 = arcpy.env.snapRaster
arcpy.env.snapRaster = Select Habitat from map
tempEnvironment2 = arcpy.env.extent
```

```
arcpy.env.extent = Select Connectivity Files

tempEnvironment3 = arcpy.env.cellSize

arcpy.env.cellSize = Select Connectivity Files

tempEnvironment4 = arcpy.env.mask

arcpy.env.mask = Mask

arcpy.gp.WeightedSum_sa(Select_Connectivity_Files,
   v_NameArea_HabitatName_Connectivity_Distance_LARCHSCAN_RAW_tif)

arcpy.env.outputCoordinateSystem = tempEnvironment0

arcpy.env.snapRaster = tempEnvironment1

arcpy.env.extent = tempEnvironment2

arcpy.env.cellSize = tempEnvironment3

arcpy.env.mask = tempEnvironment4


# Process: 0 (2)

# Process: Mosaic To New Raster

tempEnvironment0 = arcpy.env.mask

arcpy.env.mask = Mask

arcpy.MosaicToNewRaster_management("%OutputFolder%\\tmp_0.tif", OutputFolder,
   "tmp_clust3.tif", Spatial_Reference_for_Raster, "16_BIT_UNSIGNED", "", "1", "LAST",
   "FIRST")

arcpy.env.mask = tempEnvironment0


# Process: Int

tempEnvironment0 = arcpy.env.outputCoordinateSystem

arcpy.env.outputCoordinateSystem = Spatial Reference for Raster

arcpy.gp.Int_sa(Mosaic, IntMosaic)

arcpy.env.outputCoordinateSystem = tempEnvironment0


# Process: 0

# Process: Con

tempEnvironment0 = arcpy.env.outputCoordinateSystem

arcpy.env.outputCoordinateSystem = Spatial Reference for Raster

arcpy.gp.Con_sa(v_NameArea_HabitatName_Connectivity_Distance_LARCHSCAN_RAW_tif,
   Input_true_raster_or_constant_value, tmp_clust1_tif, tmp_nodata_tif, Threshold)

arcpy.env.outputCoordinateSystem = tempEnvironment0


# Process: Region Group

tempEnvironment0 = arcpy.env.outputCoordinateSystem

arcpy.env.outputCoordinateSystem = Spatial Reference for Raster

arcpy.gp.RegionGroup_sa(tmp_clust1_tif, v_NameArea_HabitatName_HabitatClusters_tif, "EIGHT",
   "WITHIN", "NO_LINK", "")

arcpy.env.outputCoordinateSystem = tempEnvironment0
```

```
# Process: Reclassify
tempEnvironment0 = arcpy.env.outputCoordinateSystem
arcpy.env.outputCoordinateSystem = Spatial Reference for Raster
tempEnvironment1 = arcpy.env.snapRaster
arcpy.env.snapRaster = Select Habitat from map
tempEnvironment2 = arcpy.env.cellSize
arcpy.env.cellSize = Select Habitat from map
tempEnvironment3 = arcpy.env.mask
arcpy.env.mask = Mask
arcpy.gp.Reclassify_sa(Select_Habitat_from_map, "Value", Select_Habitatclasses,
   v_NameArea_HabitatName_HabitatQual_tif, "NODATA")
arcpy.env.outputCoordinateSystem = tempEnvironment0
arcpy.env.snapRaster = tempEnvironment1
arcpy.env.cellSize = tempEnvironment2
arcpy.env.mask = tempEnvironment3

# Process: Zonal Statistics as Table
arcpy.gp.ZonalStatisticsAsTable_sa(v_NameArea_HabitatName_HabitatClusters_tif, "VALUE",
   v_NameArea_HabitatName_HabitatQual_tif, HabQual_sumtable, "DATA", "SUM")

# Process: Build Raster Attribute Table
arcpy.BuildRasterAttributeTable_management(v_NameArea_HabitatName_HabitatClusters_tif,
   "Overwrite")

# Process: Add Field
arcpy.AddField_management(HabQual_sumtable, "Habsize", "DOUBLE", "", "2", "", "",
   "NULLABLE", "NON_REQUIRED", "")

# Process: Calculate Field
arcpy.CalculateField_management(clust2, "Habsize", "(((%CellSize_m% * %CellSize_m%) / 10000)
   /100)* [Sum]", "VB", "")

# Process: Join Field
arcpy.JoinField_management(v_NameArea_HabitatName_HabitatClusters_tif, "Value", clust3,
   "Value", "Habsize")

# Process: Lookup
arcpy.gp.Lookup_sa(tmp_clustjoin, "Habsize",
   v_NameArea_HabitatName_Connectivity_Distance_HabSizeHa_tif)
```