**8. Appendices**

**Appendix 1: Table showing bands selected as classification predictors.**

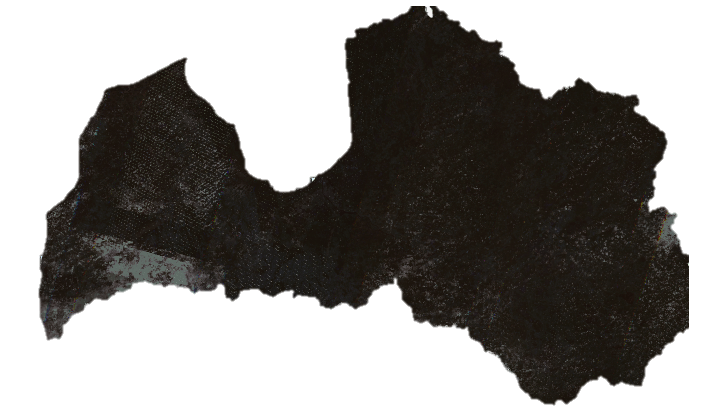
|  |  |  |
| --- | --- | --- |
| **Band** | **Wavelength (μm)** | **Description** |
| **B1** | 0.45-0.52 | Band 1 (blue) surface reflectance |
| **B2** | 0.52-0.60 | Band 2 (green) surface reflectance |
| **B3** | 0.62-0.69 | Band 3 (red) surface reflectance |
| **B4** | 0.77-0.90 | Band 4 (near infrared) surface reflectance |
| **B5** | 1.55-1.75 | Band 5 (shortwave infrared 1) surface reflectance |
| **B7** | 2.08-2.35 | Band 7 (shortwave infrared 2) surface reflectance |

**Appendix 2: Criteria chosen to represent each additional land use class.**

|  |  |
| --- | --- |
| **Land use** | **Criteria** |
| **Forestry** | Filtered to include all land used for forestry, including the production of timber, firewood and round wood. |
| **Wetlands** | Filtered to include all land classed as areas that fall between land and water, usually being inundated with water on a temporary or permanent basis. |
| **Water** | Filtered to include all land classed as water, including inland and coastal areas without vegetation that are covered by water. |
| **Artificial** | Filtered to include all land classed as artificial, including built-up areas and humanmade areas characterised by materials like concrete and gravel. |

**Appendix 3: Total number of points obtained from LUCAS** and the number used for training (80%) and testing (20%) for additional classes. Total includes three key classes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Land type** | **Number of LUCAS points** | **Number of points in training sample** | **Number of points in testing sample** |
| **Forestry** | 2272 | 1815 | 457 |
| **Wetlands** | 98 | 132 | 35 |
| **Water** | 167 | 1038 | 257 |
| **Artificial** | 69 | 54 | 15 |
| **Total** | 4013 | 3223 | 817 |



**Appendix 4: Map indicating Landsat 7 error**, as seen by the vertical lines in northwest Latvia. Image created in GEE. Scale of 50 kilometres.

**Appendix 5: Classification accuracy and error**

My classifier had a resubstitution accuracy of 96.9%, with the majority of pixels being redistributed to the correct class. The additional cloud mask increased average classification accuracy across all study years by 4%. My classifier had an average test accuracy of 70%.

**Appendix 5.1: Test accuracy for each year and the average across all years.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Year** | **Test Accuracy** | **Year** | **Test Accuracy** |
| **1989** | 0.71 | **2001** | 0.70 |
| **1990** | 0.69 | **2002** | 0.70 |
| **1991** | 0.70 | **2003** | 0.67 |
| **1992** | 0.70 | **2004** | 0.67 |
| **1993** | 0.69 | **2005** | 0.69 |
| **1994** | 0.72 | **2006** | 0.70 |
| **1995** | 0.70 | **2007** | 0.71 |
| **1996** | 0.71 | **2008** | 0.68 |
| **1997** | 0.73 | **2009** | 0.70 |
| **1998** | 0.69 | **2010** | 0.69 |
| **1999** | 0.71 | **2011** | 0.72 |
| **2000** | 0.71 | **Average** | 0.70 |

**Appendix 5.2: Conceptual output of confusion matrix.**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Predicted Class A** | **Predicted Class B** | **Predicted Class C** |
| **Known Class A** | **True A** | False B | False C |
| **Known Class B** | False A | **True B** | False C |
| **Known Class C** | False A | False B | **True C** |

**Appendix 5.3: Example confusion matrix for 2011, which is the year with the highest test accuracy**. Bolded numbers represent the number of points classed correctly. Other values are classed incorrectly.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Forestry** | **Abandoned** | **Extensive** | **Intensive** | **Water** | **Wetlands** | **Artificial** |
| **Forestry** | **19688** | 8 | 1 | 3459 | 61 | 27 | 8 |
| **Abandoned** | 548 | **0** | 0 | 526 | 1 | 0 | 0 |
| **Extensive** | 141 | 0 | **0** | 312 | 1 | 0 | 0 |
| **Intensive** | 2997 | 12 | 1 | **9986** | 41 | 6 | 31 |
| **Water** | 1063 | 2 | 0 | 372 | **344** | 1 | 0 |
| **Wetlands** | 862 | 0 | 0 | 252 | 6 | **8** | 0 |
| **Artificial** | 280 | 0 | 0 | 489 | 33 | 0 | **10** |

From this, it is clear that my classifier overestimated forestry and intensive land use cover. It incorrectly classed abandoned and extensive land cover 100% of the time.

**Appendix 6: Results in tabular form and additional results**

**Appendix 6.1** – **Abandoned and extensive land cover decreased, and intensive land cover increased directly following SUC**. With a low standard error and the highest marginal R2, abandoned LUC shows the strongest relationship with SUC (abandoned – N = 255; extensive – N = 247; intensive – N = 258).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Land use** | **Estimate** | | **± SE of estimate** | **t-value** | | ***p*-value** | | **R2M** | **R2C** |
| **Abandoned** | -5.14 | 1.13 | | | -4.56 | | <0.0001 | 0.06 | 0.30 |
| **Extensive** | -0.08 | 0.13 | | | -0.60 | | 0.55 | <0.01 | 0.01 |
| **Intensive** | 2089.51 | 2605.67 | | | 0.80 | | 0.42 | <0.01 | 0.11 |

**Appendix 6.2** – **Abandoned and extensive land cover decreased, and intensive land cover increased directly following EUA**. With a low standard error relative to the slope and the highest marginal R2, intensive LUC shows the strongest relationship with EUA (abandoned – N = 255; extensive – N = 247; intensive – N = 258).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Land use** | **Estimate** | | **± SE of estimate** | **t-value** | | ***p*-value** | | **R2M** | **R2C** |
| **Abandoned** | -2.61 | 1.20 | | | -2.12 | | 0.03 | 0.01 | 0.10 |
| **Extensive** | -0.01 | 0.09 | | | -0.13 | | 0.90 | <0.0001 | 0.05 |
| **Intensive** | 6179.11 | 1942.18 | | | 3.18 | | <0.01 | 0.04 | 0.09 |

**Appendix 6.3** – Examining the impact of location as a predictor of LUC

**Appendix 6.3.1** – **Models including cell as a random effect show less error and increased model fit (R2C).**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Event** | **Land use** | **Estimate** | **± SE of estimate** | **t-value** | ***p*-value** | **R2M** | **R2C** |
| **SUC** | **Abandoned** | -5.11 | 0.68 | -7.47 | <0.0001 | 0.04 | 0.48 |
| **Extensive** | -0.09 | 0.06 | -1.39 | 0.16 | <0.01 | 0.52 |
| **Intensive** | 2089.51 | 331.90 | 6.30 | <0.0001 | <0.01 | 0.96 |
| **EUA** | **Abandoned** | -2.59 | 0.85 | -3.05 | <0.01 | <0.01 | 0.29 |
| **Extensive** | -0.02 | 0.04 | -0.50 | 0.61 | <0.01 | 0.54 |
| **Intensive** | 6204.74 | 620.31 | 10.00 | <0.0001 | 0.03 | 0.77 |

A close up of a map

Description automatically generated

**a.**

**b.**

**Appendix 6.3.2 –** **Figure depicting average abandoned land area per year per cell (a) aggregated by grid and (b) shown for all cells increasing in abandoned land cover**. Scale of y-axis different for effective visualisation – two graphs should not be compared directly. Raw data used solely for visualisation purposes.

**Appendix 6.4** – **The amount of land transitioning to abandoned and extensive land from intensive land decreased following SUC**. The direction for the transition to extensive land is likely arbitrary due to the large error relative to the slope (N = 258).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Land use transition** | **Estimate** | | **± SE of estimate** | **t-value** | | ***p*-value** | | **R2M** | **R2C** |
| **Intensive to**  **abandoned** | -3.02 | 0.70 | | | -3.76 | | <0.001 | 0.04 | 0.24 |
| **Intensive to extensive** | -0.03 | 0.04 | | | -0.67 | | 0.50 | <0.01 | 0.05 |

**Appendix 6.5** – **The amount of land transitioning to from abandoned to intensive decreased following EUA, whereas land transitioning to intensive from extensive increased following SUC**. Standard error is high relative to the slope and the model fit is low for both relationships, indicating a lack of a clear relationship (N = 258).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Land use transition** | **Estimate** | | **± SE of estimate** | **t-value** | | ***p*-value** | | **R2M** | **R2C** |
| **Abandoned to intensive** | -0.40 | 0.78 | | | -0.51 | | 0.61 | <0.01 | 0.05 |
| **Extensive to intensive** | 0.04 | 0.04 | | | 1.12 | | 0.27 | <0.01 | 0.02 |

**Appendix 6.6** – **Unclear relationships tested on LUC six years following EUA.** No clear relationship seen, owing to large error relative to the slope and low model fit (abandoned – N = 255; extensive – N = 249; A 🡪 I – N = 258, E 🡪 I – N = 258).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Land use transition** | **Estimate** | | **± SE of estimate** | **t-value** | | ***p*-value** | | **R2M** | **R2C** |
| **Abandoned** | -2.07 | 1.21 | | | -1.73 | | 0.09 | 0.01 | 0.13 |
| **Extensive** | -0.01 | 0.08 | | | -0.18 | | 0/86 | <0.001 | 0.10 |
| **Abandoned to intensive** | 0.04 | 0.04 | | | 1.12 | | 0.27 | <0.001 | 0.02 |
| **Extensive to intensive** | 0.04 | 0.03 | | | 1.11 | | 0.27 | <0.001 | 0.12 |

**A screenshot of a cell phone

Description automatically generatedA close up of text on a white background

Description automatically generated**

**a.**

**b.**

**Appendix 6.7.1 – Breakpoint figures for (a) the transition from intensive to abandoned land and (b) the transition from abandoned to intensive land**. Coloured line shows total land use area (km2) per year. Dotted line represents the two SPE events (SUC and EUA). Dashed line indicates where model fit two breakpoints. Solid black line represents segmented regression, as fit by model.

**A close up of a map

Description automatically generatedA screenshot of a cell phone

Description automatically generated**

**a.**

**b.**

**Appendix 6.7.2 – Breakpoint figures for (a) the transition from extensive to abandoned land and (b) the transition from extensive to intensive land**. Coloured line shows total land use area (km2) per year. Dotted line represents the two SPE events (SUC and EUA). Dashed line indicates where model fit two breakpoints. Solid black line represents segmented regression, as fit by model.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Land use** | **Breakpoint 1** | | **Error** | | **Breakpoint 2** | **Error** | |
| **Abandoned** | 1992.8 | 2.15 | | 1994.2 | | | 2.29 | |
| **Extensive** | 1995.2 | 1.93 | | 2007.3 | | | 1.22 | |
| **Intensive** | 2002.5 | 2.15 | | 2004.2 | | | 1.91 | |
| **Intensive to abandoned** | 1993.8 | 2.35 | | 2007.4 | | | 2.72 | |
| **Intensive to extensive** | 2003.0 | 1.42 | | 2009.6 | | | 0.64 | |
| **Extensive to intensive** | 2000.7 | 1.75 | | 2002.1 | | | 2.65 | |
| **Abandoned to intensive** | 1991.9 | 0.82 | | 1993.1 | | | 1.34 | |

**Appendix 6.8: Complete list of breakpoints and their associated errors**

**Appendix 7: GEE code excerpts. For full code, for full code visit** https://github.com/izzyrich/dissertation. First box shows example code for classification and second box shows example code for calculations.

|  |
| --- |
| // classify land use in Latvia with LUCAS (2012) data  // SET UP DATA ----  // choose bands  **var** bands = ['B1', 'B2', 'B3', 'B4', 'B5', 'B7'];  // import all points of known land-use type from LUCAS  **var** fc\_total = **ee.FeatureCollection**('users/izzyrich/full\_2012');  **var** forestry = **ee.FeatureCollection**('users/izzyrich/forestry');  **var** abandoned = **ee.FeatureCollection**('users/izzyrich/abandoned');  **var** extensive = **ee.FeatureCollection**('users/izzyrich/extensive');  **var** intensive = **ee.FeatureCollection**('users/izzyrich/intensive');  **var** water = **ee.FeatureCollection**('users/izzyrich/water');  **var** wetlands = **ee.FeatureCollection**('users/izzyrich/wetlands');  **var** artificial = **ee.FeatureCollection**('users/izzyrich/artificial');  // function to cloud correct  **var** cloudMaskL457 = **function**(image) {  **var** qa = image.select('pixel\_qa');  // If the cloud bit (5) is set and the cloud confidence (7) is high  // or the cloud shadow bit is set (3), then it's a bad pixel.  **var** cloud = qa.bitwiseAnd(1 << 5)  .and(qa.bitwiseAnd(1 << 7))  .or(qa.bitwiseAnd(1 << 3))  // Remove edge pixels that don't occur in all bands  **var** mask2 = image.mask().reduce(ee.Reducer.min());  **return** image.updateMask(cloud.not()).updateMask(mask2);  };  // import shape of Latvia polygon to define ROI  **var** latvia\_poly = **ee.FeatureCollection**('users/izzyrich/latvia\_poly');  // add buffer of 90 metres to create polygons – do for each land use  **var** fc\_total = fc\_total.map(**function**(f) {  **return** f.buffer(90);  });  // add satellite imagery for 2011 - surface reflectance  **var** landsatCollection = **ee.ImageCollection**('LANDSAT/LT05/C01/T1\_SR')  .filterDate('2011-06-01', '2011-08-30');    // get median of imagery to remove high and low reflectance (cloud and shadow)  **var** median = landsatCollection.map(cloudMaskL457).median();  // clip on size of latvia  **var** clipped = median.clip(latvia\_poly);  // START CLASSIFICATION ----  // set random seed  **var** n = 0;  // get random columns for each land-use type  **var** randomForestry = forestry.randomColumn('random', n);  **var** randomAbandoned = abandoned.randomColumn('random', n);  **var** randomExtensive = extensive.randomColumn('random', n);  **var** randomIntensive = intensive.randomColumn('random', n);  **var** randomWater = water.randomColumn('random', n);  **var** randomWetlands = wetlands.randomColumn('random', n);  **var** randomArtificial = artificial.randomColumn('random', n);  // split up data for testing and training - 80% for training and 20% for testing  **var** split = 0.8;  // stratified training and testing samples  **var** trainingSample = randomForestry.filter(ee.Filter.lt('random', split))  .merge(randomAbandoned.filter(ee.Filter.lt('random', split)))  .merge(randomExtensive.filter(ee.Filter.lt('random', split)))  .merge(randomIntensive.filter(ee.Filter.lt('random', split)))  .merge(randomWater.filter(ee.Filter.lt('random', split)))  .merge(randomWetlands.filter(ee.Filter.lt('random', split)))  .merge(randomArtificial.filter(ee.Filter.lt('random', split)));    **var** testingSample = randomForestry.filter(ee.Filter.gte('random', split))  .merge(randomAbandoned.filter(ee.Filter.gte('random', split)))  .merge(randomExtensive.filter(ee.Filter.gte('random', split)))  .merge(randomIntensive.filter(ee.Filter.gte('random', split)))  .merge(randomWater.filter(ee.Filter.gte('random', split)))  .merge(randomWetlands.filter(ee.Filter.gte('random', split)))  .merge(randomArtificial.filter(ee.Filter.gte('random', split)));  // Sample the input imagery to get a FeatureCollection of training data.  **var** training = clipped.select(bands).sampleRegions({  collection: trainingSample,  properties: ['class'],  scale: 30,  });  // trained with 80% of our data  **var** trainedClassifier = ee.Classifier.randomForest({  numberOfTrees: 30  })  .train(training, 'class');  // classify FeatureCollection  **var** classified = clipped.classify(trainedClassifier, 'classification');  // Get a confusion matrix representing resubstitution accuracy.  print('RF error matrix: ', trainedClassifier.confusionMatrix());  print('RF accuracy: ', trainedClassifier.confusionMatrix().accuracy());  // Sample input to get validation data  **var** validation = clipped.sampleRegions({  collection: testingSample,  properties: ['class'],  scale: 30,  });  // Classify validation data  **var** validated = validation.classify(trainedClassifier);  // Get error of testing data + export to table to save time  **var** testError = validated.errorMatrix('class', 'classification');  **var** exportconfusionMatrix = **ee.Feature**(**null**, {matrix: testError.array()});  Export.table.toDrive({  collection: **ee.FeatureCollection**(exportconfusionMatrix),  description: 'exportconfusionMatrix\_2011',  fileFormat: 'CSV'  });  // Get accuracy of testing data + export to table to save time  **var** testAccuracy = testError.accuracy();  **var** exporttestAccuracy = **ee.Feature**(**null**, {matrix: testAccuracy});  Export.table.toDrive({  collection: **ee.FeatureCollection**(exporttestAccuracy),  description: 'testAccuracy\_2011',  fileFormat: 'CSV'  });  // Export the image to an Earth Engine asset.  Export.image.toAsset({  image: classified.select(['classification']),  description: 'classified',  assetId: 'classified',  scale: 30,  region: poly,  maxPixels: 1e13  }); |

|  |
| --- |
| // import all classification images - example  **var** classified\_1989 = **ee.Image**('users/izzyrich/classified\_1989');  **var** classified\_1990 = **ee.Image**('users/izzyrich/classified\_1990');  // import shape of Latvia polygon to define ROI  **var** latvia\_poly = **ee.FeatureCollection**('users/izzyrich/latvia\_poly');  // MAKE GRID FOR Latvia  // 1) Create bounding box  **var** lon\_start = 20.97139;  **var** lon\_end = 29.24051;  **var** lat\_start = 55.66372;  **var** lat\_end = 58.08577;  // 2) Decide no. of (in this case: equally sized) cells  **var** num\_cells = 200;  **var** lon\_edge = (lon\_end-lon\_start)/Math.sqrt(num\_cells);  **var** lat\_edge = (lat\_end-lat\_start)/Math.sqrt(num\_cells);  // 3) Create the grid  **var** polys = [];  **var** polys\_line = [];  **var** cell\_id = 0;  **for** (**var** lon = lon\_start; lon < lon\_end; lon += lon\_edge) {  **var** x1 = lon;  **var** x2 = lon + lon\_edge;  **for** (**var** lat = lat\_start; lat < lat\_end; lat += lat\_edge) {  cell\_id = cell\_id + 1;  **var** y1 = lat;  **var** y2 = lat + lat\_edge;  polys.push(**ee.Feature**(**ee.Geometry.Rectangle**(x1, y1, x2, y2), {label: cell\_id}));  }  }  **var** grid = **ee.FeatureCollection**(polys);  **var** filtered = grid.filterBounds(latvia\_poly);  // calculate transition area  **var** transitionarea = **function**(image1, image2, class1, class2, name){  **var** first = image1.select('classification').eq(class1);  **var** second = image2.select('classification').eq(class2);  **var** change = first.and(second);  **var** reduce = change.addBands(change).reduceRegions({  collection: filtered,  reducer: ee.Reducer.sum().group({  groupField: 1,  groupName: 'classification',  }),  scale:30});  **var** feature = **ee.Feature**(reduce);  Export.table.toDrive({  collection: feature,  description: name,  fileFormat: 'CSV'  });  };  transitionarea(classified\_1989, classified\_1990, 1, 3, '89\_90\_1to3');  transitionarea(classified\_1989, classified\_1990, 1, 2, '89\_90\_1to2');  transitionarea(classified\_1989, classified\_1990, 2, 1, '89\_90\_2to1');  transitionarea(classified\_1989, classified\_1990, 2, 3, '89\_90\_2to3');  transitionarea(classified\_1989, classified\_1990, 3, 1, '89\_90\_3to1');  transitionarea(classified\_1989, classified\_1990, 3, 2, '89\_90\_3to2');  // calculate area of each class  **var** area = **function**(image, name){  **var** areacount = image.addBands(image).reduceRegions({  collection: filtered,  reducer: ee.Reducer.sum().group({  groupField: 1,  groupName: 'classification',  }),  scale:30});  **var** reduce = **ee**.**FeatureCollection**(**ee**.**Feature**(areacount));  Export.table.toDrive({  collection: areacount,  description: name,  fileFormat: 'CSV'  });  };  area(classified\_1989, 'classified\_1989'); |

**Appendix 8: R code excerpts. For full code, for full code visit** https://github.com/izzyrich/dissertation.

**Packages used for all code**

|  |
| --- |
| *# Load packages* library(readr)library(tidyverse)library(rgdal)library(sp)library(raster)library(sf)library(rworldmap)library(grid)library(rworldxtra)library(stringr)library(maptools)library(tiff)library(dggridR)library(modeest)library(lme4)library(MuMIn)library(lmerTest)library(segmented)library(effects)library(scales)library(gridExtra) |

**Filter and format LUCAS data points and create polygon of Latvia**

|  |
| --- |
| # Import base dataset  lucas <- read\_csv("data/2012\_lucas.csv")  # LATVIA BORDER and POLY ----  # get data  latvia <- raster::getData("GADM", country = "LVA", level = 0)  # change projection latvia <- spTransform(latvia, "+proj=longlat +ellps=WGS84 +datum=WGS84 +init=epsg:3857")  # create polygon data = data.frame(f=99.9) spdf = SpatialPolygonsDataFrame(latvia, data) shapefile(spdf, "data/latvia\_poly", overwrite = TRUE)  # format data map <- fortify(latvia)%>%  dplyr::select(long, lat)  colnames(map)[colnames(map) == "lat"] <- "LAT" colnames(map)[colnames(map) == "long"] <- "LONG"  # set CRS and transform coordinates(map) <- c("LONG", "LAT") proj4string(map) <- CRS("+proj=longlat +ellps=WGS84 +datum=WGS84 +init=epsg:3857") border\_points <- spTransform(map, "+proj=longlat +ellps=WGS84 +datum=WGS84 +init=epsg:3857") r\_border <- raster(border\_points) # border as raster  ## ABANDONED LAND ----  # filter for abandoned land U112\_options <- c("10", "20") # stated that these were abandoned agricultural areas U410\_options <- c("B", "C", "D", "E", "F") # classes that could be agriculture in U410 (natural terrestrial areas)  lucas\_filtered <- lucas %>%   separate(LC1, into = c('class', 'number'), sep = 1) %>%  filter(LU1 == "U410" & class %in% U410\_options |  LU1 == "U112" & class == "D" & number %in% U112\_options |  LU1 == "U420" & class == "E" & number == "30") %>%  dplyr::select(GPS\_LAT, GPS\_LONG) %>% # potentially need elevation here??  mutate(class = "1") %>%  mutate(name = "abandoned")  # write to csv  write.csv(lucas\_filtered, file = "data/lucas\_2012\_filtered.csv") |

**Format and aggregate all data from classification**

|  |
| --- |
| # TOTAL AREA ----  # import data 1989 ---- data <- read\_csv("~/Documents/Edinburgh Year 4/dissertation/data/classified\_1989.csv")  # format ---- new <- data %>%  separate(groups, into = c('class0', 'number0',   'class1', 'number1',   'class2', 'number2',   'class3', 'number3',   'class4', 'number4',   'class5', 'number5',   'class6', 'number6'), sep = ",")   new$class1 <- gsub("[^0-9.-]", "", new$class1) new$number1 <- gsub("[^0-9.-]", "", new$number1) new$class2 <- gsub("[^0-9.-]", "", new$class2) new$number2 <- gsub("[^0-9.-]", "", new$number2) new$class3 <- gsub("[^0-9.-]", "", new$class3) new$number3 <- gsub("[^0-9.-]", "", new$number3)  new$number1 <- as.numeric(new$number1) new$number2 <- as.numeric(new$number2) new$number3 <- as.numeric(new$number3)   delete89 <- new %>%  dplyr::select(-c(.geo, `system:index`))  new\_df89 <- bind\_rows(  delete89 %>% dplyr::select(label, class = class1, pixels = number1),  delete89 %>% dplyr::select(label, class = class2, pixels = number2),  delete89 %>% dplyr::select(label, class = class3, pixels = number3), ) %>%  mutate(area = pixels\*30) %>%  mutate(year = 1989)  # TOTAL TRANSITION ----  # import data 89 to 90 1 to 3 ---- data <- read\_csv("~/Documents/Edinburgh Year 4/dissertation/data/89\_90\_1to3.csv")  # format ---- new <- data %>%  separate(groups, into = c('class0', 'number0',   'class1', 'number1'), sep = ",")   new$class0 <- gsub("[^0-9.-]", "", new$class0) new$number0 <- gsub("[^0-9.-]", "", new$number0) new$class1 <- gsub("[^0-9.-]", "", new$class1) new$number1 <- gsub("[^0-9.-]", "", new$number1)  new$number0 <- as.numeric(new$number0) new$number1 <- as.numeric(new$number1)  transition\_8990\_1 <- new %>%  dplyr::select(-c(.geo, `system:index`, class0, class1, number0)) %>%  mutate(area = number1\*30) %>%  mutate(year = "1990") %>%  mutate(transition = "1") %>%  mutate(previous\_class = "1") %>%  mutate(current\_class = "3") |

**Create large grid and conduct statistical analyses**

|  |
| --- |
| # load data ---- detailed\_area <- read\_csv("data/detailed\_area.csv") %>%  dplyr::select(-c("X1")) colnames(detailed\_area)[colnames(detailed\_area) == "label"] <- "cell" detailed\_area$class <- factor(detailed\_area$class) detailed\_area$cell <- factor(detailed\_area$cell)  detailed\_transition <- read\_csv("data/detailed\_transition.csv") %>%  dplyr::select(-c("X1")) colnames(detailed\_transition)[colnames(detailed\_transition) == "label"] <- "cell" detailed\_transition$cell <- factor(detailed\_transition$cell) detailed\_transition$transition <- factor(detailed\_transition$transition)  # apply bigger grid detailedA <- detailed\_area %>%  mutate(grid = if\_else(cell == 8 | cell == 9 | cell == 10 | cell == 11 | cell == 23 |  cell == 24 | cell == 25 | cell == 26 | cell == 27 |  cell == 38 | cell == 39 | cell == 40 | cell == 41 |  cell == 42 | cell == 43 | cell == 53 | cell == 54 | cell == 55 | cell == 56, "NW",   if\_else(cell == 3 | cell == 4 | cell == 5 | cell == 6 | cell == 7 |  cell == 19 | cell == 20 | cell == 21 | cell == 22 |  cell == 35 | cell == 36 | cell == 37 | cell == 49 |  cell == 50 | cell == 51 | cell == 52, "SW",   if\_else(cell == 64 | cell == 65 | cell == 66 | cell == 67 | cell == 68 |  cell == 69 | cell == 79 | cell == 80 | cell == 81 |  cell == 82 | cell == 83 | cell == 84 | cell == 85 |  cell == 86 | cell == 87 | cell == 88 | cell == 89 | cell == 94 |   cell == 95 | cell == 96 | cell == 97 | cell == 98 | cell == 99 |  cell == 100 | cell == 101 | cell == 102 | cell == 103 | cell == 104, "C",  if\_else(cell == 113 | cell == 114 | cell == 115 | cell == 116 | cell == 117 |  cell == 118 | cell == 119 | cell == 120 | cell == 128 |  cell == 129 | cell == 130 | cell == 131 | cell == 132 |  cell == 133 | cell == 134 | cell == 143 | cell == 144 | cell == 145 |   cell == 146 | cell == 147 | cell == 158 | cell == 159 | cell == 160 |  cell == 161 | cell == 162 | cell == 173 | cell == 174 | cell == 175 | cell == 176, "NE",  if\_else(cell == 108 | cell == 109 | cell == 110 | cell == 111 | cell == 112 |  cell == 121 | cell == 122 | cell == 123 | cell == 124 |  cell == 125 | cell == 126 | cell == 127 | cell == 136 |  cell == 137 | cell == 138 | cell == 139 | cell == 140 | cell == 141 |   cell == 142 | cell == 151 | cell == 152 | cell == 153 | cell == 154 |  cell == 155 | cell == 156 | cell == 157 | cell == 166 | cell == 167 | cell == 168 |  cell == 169 | cell == 170 | cell == 171 | cell == 172 | cell == 183 | cell == 184 | cell == 185 |  cell == 186, "SE", "NA"))))))   # apply bigger grid detailedT <- detailed\_transition %>%  mutate(grid = if\_else(cell == 8 | cell == 9 | cell == 10 | cell == 11 | cell == 23 |  cell == 24 | cell == 25 | cell == 26 | cell == 27 |  cell == 38 | cell == 39 | cell == 40 | cell == 41 |  cell == 42 | cell == 43 | cell == 53 | cell == 54 | cell == 55 | cell == 56, "NW",   if\_else(cell == 3 | cell == 4 | cell == 5 | cell == 6 | cell == 7 |  cell == 19 | cell == 20 | cell == 21 | cell == 22 |  cell == 35 | cell == 36 | cell == 37 | cell == 49 |  cell == 50 | cell == 51 | cell == 52, "SW",   if\_else(cell == 64 | cell == 65 | cell == 66 | cell == 67 | cell == 68 |  cell == 69 | cell == 79 | cell == 80 | cell == 81 |  cell == 82 | cell == 83 | cell == 84 | cell == 85 |  cell == 86 | cell == 87 | cell == 88 | cell == 89 | cell == 94 |   cell == 95 | cell == 96 | cell == 97 | cell == 98 | cell == 99 |  cell == 100 | cell == 101 | cell == 102 | cell == 103 | cell == 104, "C",  if\_else(cell == 113 | cell == 114 | cell == 115 | cell == 116 | cell == 117 |  cell == 118 | cell == 119 | cell == 120 | cell == 128 |  cell == 129 | cell == 130 | cell == 131 | cell == 132 |  cell == 133 | cell == 134 | cell == 143 | cell == 144 | cell == 145 |   cell == 146 | cell == 147 | cell == 158 | cell == 159 | cell == 160 |  cell == 161 | cell == 162 | cell == 173 | cell == 174 | cell == 175 | cell == 176, "NE",  if\_else(cell == 108 | cell == 109 | cell == 110 | cell == 111 | cell == 112 |  cell == 121 | cell == 122 | cell == 123 | cell == 124 |  cell == 125 | cell == 126 | cell == 127 | cell == 136 |  cell == 137 | cell == 138 | cell == 139 | cell == 140 | cell == 141 |   cell == 142 | cell == 151 | cell == 152 | cell == 153 | cell == 154 |  cell == 155 | cell == 156 | cell == 157 | cell == 166 | cell == 167 | cell == 168 |  cell == 169 | cell == 170 | cell == 171 | cell == 172 | cell == 183 | cell == 184 | cell == 185 |  cell == 186, "SE", "NA"))))))   # DF for Q1 # before and after in km2 per cell  questiononeSUC <- detailedA %>%  dplyr::filter(year == 1989 | year == 1990 | year == 1991 | year == 1992 | year == 1993 | year == 1994) %>%  dplyr::select(-c(pixels)) %>%  mutate(before\_after = ifelse(year == 1989 | year == 1990 | year == 1991, "first", "second")) %>%  dplyr::select(-c(year)) %>%  group\_by(before\_after, grid, class, cell) %>%  summarise(area = mean(area)/1000)   # DF for Q2 # before and after in km2 questiontwoSUC <- detailedT %>%  dplyr::filter(year == 1990 | year == 1991 | year == 1992 | year == 1993 | year == 1994 | year == 1995) %>%  dplyr::select(-c(pixels)) %>%  mutate(before\_after = ifelse(year == 1990 | year == 1991 | year == 1992, "first", "second")) %>%  dplyr::select(-c(year)) %>%  group\_by(before\_after, grid, transition, cell) %>%  summarise(area = mean(area)/1000)    # DF for Q3 ---- questionthreeASUC <- detailedA %>%  dplyr::filter(year == 1989 | year == 1990 | year == 1991 | year == 1995 | year == 1996 | year == 1997) %>%  dplyr::select(-c(pixels)) %>%  mutate(before\_after = ifelse(year == 1989 | year == 1990 | year == 1991, "first", "second")) %>%  dplyr::select(-c(year)) %>%  group\_by(before\_after, grid, class, cell) %>%  summarise(area = mean(area)/1000)  # Abandoned SUC ---- questiononeSUC1 <- questiononeSUC %>%  dplyr::filter(class == 1)   questiononeSUC1$grid <- factor(questiononeSUC1$grid) questiononeSUC1$cell <- factor(questiononeSUC1$cell) questiononeSUC1$before\_after <- factor(questiononeSUC1$before\_after)  # model abandonedSUC <- lmer(area ~ before\_after + (1|grid), data = questiononeSUC1) summary(abandonedSUC)  r.squaredGLMM(abandonedSUC)  # Q2 A--I: EUA ---- questiontwoEUA1 <- questiontwoEUA %>%  dplyr::filter(transition == 1)  questiontwoEUA1$grid <- factor(questiontwoEUA1$grid) questiontwoEUA1$cell <- factor(questiontwoEUA1$cell) questiontwoEUA1$before\_after <- factor(questiontwoEUA1$before\_after)  atoiEUA <- lmer(area ~ before\_after + (1|grid), data = questiontwoEUA1) summary(atoiEUA)  r.squaredGLMM(atoiEUA)  # Q3 lag Abandoned SUC ---- questionthreeSUC1 <- questionthreeASUC %>%  dplyr::filter(class == 1)   questionthreeSUC1$grid <- factor(questionthreeSUC1$grid) questionthreeSUC1$cell <- factor(questionthreeSUC1$cell) questionthreeSUC1$before\_after <- factor(questionthreeSUC1$before\_after)  # model abandonedlagSUC <- lmer(area ~ before\_after + (1|grid), data = questionthreeSUC1) summary(abandonedlagSUC)  r.squaredGLMM(abandonedlagSUC)  # Q3 part b ---- abandonedseglag <- detailedA %>%  dplyr::select(-c(pixels)) %>%  filter(class == "1") %>%  group\_by(year) %>%  summarise(year\_total = sum(area)/1000)  abandonedlm <- lm(year\_total ~ year, data = abandonedseglag)  summary(abandonedlm)  abandonedmod <- segmented(abandonedlm, seg.Z = ~year, psi = list(year = c(1996,2004))) summary(abandonedmod) |