FSVM: Extracting QuadPolyID

for Training Field Data and Predicting over Extents

Robert Ritson, WMI Research Associate

2021-07-29

Extracting QuadPolyIDs for Field Data and Extents

Overview

An essential part of the fine scale vegetation modeling process is associating the eCognition polygon covariates developed by the University of Idaho with the field data collected by Idaho Department of Fish and Game and others. Both the covariate polygons and field data need to be in Idaho Transverse Mercator projection. Given the irregular shape and dimensions of the eCognition polygons, it is easier to query the polygons by their 'QuadPolyID' and join them to the field data than overlaying the field data and extracting the covariate values. But first, we need to associate the field data with a 'QuadPolyID' so we know which covariates they need to be joined to.

The method we developed uses a Python script with the ESRI 'arcpy' module to execute a spatial intersect with a shapefile of the field data or extent over a geodatabase containing the eCognition polygon shapefiles. Although the process is simple, it is extremely intensive to process due to the size of the eCognition shapefile geodatabase. This process is executed in R with the `py_extract_quadpolyID` function in the `fsvm` package with the help of the Python interpreter package `reticulate`. Due to certain software requirements for ArcGIS, this function must be run in a 32-bit architecture R session (see below, 'Switching between Architectures in R').

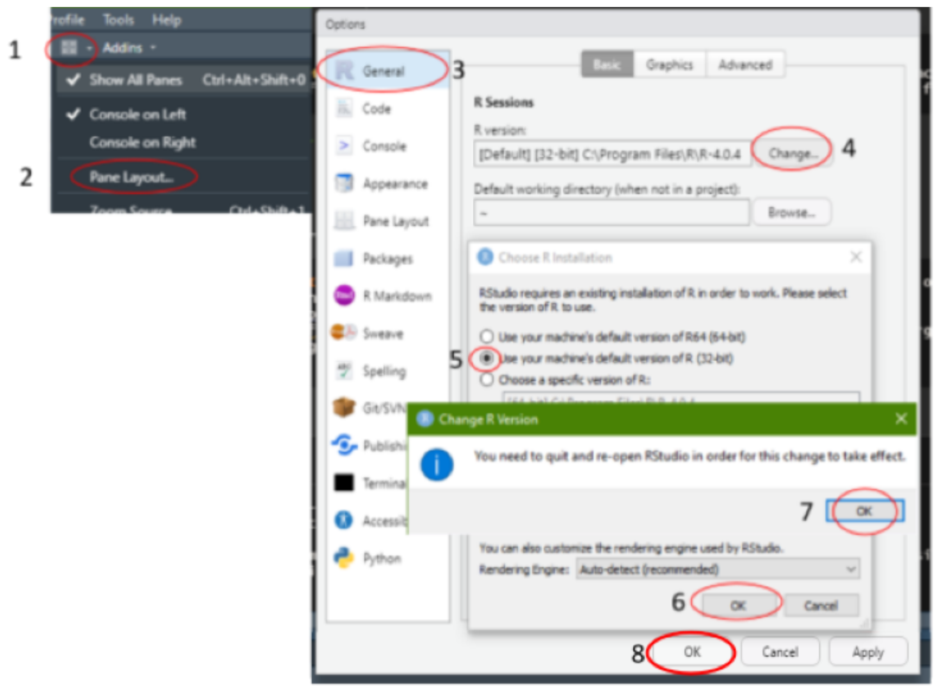
Switching between Architectures in R

To check the architecture of your current session, run the following command.

```
Sys.getenv("R_ARCH")
> [1] "/i386"
```

If "/i386" is returned, then you are currently running R in a 32-bit architecture and are ready to proceed. If the command returned "/x64", then your current R session is running in a 64-bit architecture.

To switch to 32-bit, follow these steps:



R Sessions

R version: [Default] [32-bit] C:\Program Files\R\R-4.0.4

Default working directory (when not in a project): ~

Choose R Installation

RStudio requires an existing installation of R in order to work. Please select the version of R to use.

- Use your machine's default version of R64 (64-bit)
- [✓ Use your machine's default version of R (32-bit)]
- Choose a specific version of R:

Change R Version

You need to quit and re-open RStudio in order for this change to take effect.

You can also customize the rendering engine used by RStudio.

Rendering Engine: Auto-detect (recommended)

1. Open 'Workspace Panes'
2. Go to 'Pane Layout'
3. Click 'General'
4. Click 'Change' under R version
5. In the 'Choose R Installation' window, click 'Use your machine's default version of R (32-bit)'
6. Click 'OK' in the 'Choose R Installation' window
7. In the pop-up window 'Change R Version', click 'OK'
8. Click 'OK' in the 'Options' window
9. Quit RStudio session (Ctrl+Q), then re-open RStudio

When you re-run the command `Sys.getenv("R_ARCH")` it should now return "/i386".

You are now ready to proceed with extracting QuadPolyIDs!

After completing QuadPolyID extractions, it is best to repeat the above steps and resume a 64-bit R session.

The `py_exctract_quadpolyID` function

The geodatabase of eCognition polygons contains ~40 million individual polygons stored with in 75 separate 100k USGS quadrangles which totals more than one terabyte of data. This makes executing even the most simple of functions a slow process when it comes to working with data of this size, so patience is a must. Expect to leave this function to run for several hours (or overnight) when processing large sets of field data.

The function itself carries out a relatively simple job, but requires a number of file path inputs for accessing and storing temporary files which need to be user-defined. The number of inputs can be intimidating while setting up the function, however, these parameters allow for greater flexibility and utility.

```
### NOT RUN
#`py_extract_quadpolyID` function parameters
fsvm::py_extract_quadpolyID(
py.path = "C:/Python27/ArcGIS10.6/python.exe",
fielddata.path = "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples/fsvm_dummy.shp",
quad.path = "A:/Fine scale vegetation analysis/dbases_4modeling/blank_polys100k.gdb",
output.gdb.path = "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples",
output.folder.path = "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples/Output",
output.RData = "None",
intercept.feature = "None",
newgdb.name = "Output.gdb",
quad.sel = "None",
export = T,
survey = T
)
### END NOT RUN
```

The following parameters are required arguments for associating quadpolyIDs:

- `py.path` A file path leading to Python installation you wish to use. The version of Python this file path leads to must have access to the 'arcpy' module, which generally means it is best to use the version installed by ArcGIS. The file path will differ slightly for machines using ArcPro instead of ArcGIS 10.6. For machine using ArcGIS 10.6, the default can be left as is.
- `fielddata.path` A file path leading to a shapefile or geodatabase of an input, either field data or an extent (must convert if in a csv).
- `quad.path` A file path leading to the geodatabase containing blank eCognition polygons. By default, the function looks for this on the 'HQWILDSTAT' network drive ("A:/") which means your machine must have access to this drive in order for it to run. If you have this geodatabase copied to a different location on a personal machine, you can specify it with this parameter.
- `output.gdb.path` A file path to a valid geodatabase. If you need output stored in a new geodatabase, you must provide a name to the 'newgdb.name' parameter.
  - *Troubleshooting Tip*: Errors often stem from issues with closing or opening the geodatabase. If the function fails at first (due to misspellings, etc.), it is a best practice to save your session and completely restart R. You may also need to delete the new geodatabase or switch your 'output.gdb.path' to navigate to the geodatabase you created. When the function runs successfully, this temporary geodatabase should automatically be deleted by the function.
- `output.folder.path` A file path to an output folder to store .dbfs of your intermediate output. The folder does not need to exist, the function will create the folder you name in the file path.
  - *Troubleshooting Tip*: While the Python script powering this function is supposed to overwrite files in the output folder and geodatabase, it may be necessary to delete the function created folder and geodatabase and allow them to be recreated from scratch. When the function runs successfully, this temporary folder should automatically be deleted by the function.

There are also several optional arguments for dealing with different types of field data formats:

- `output.RData` A name of for the output .RData file. The default of 'None' automatically populates the name as "FieldDataPoints_QuadPolyID.RData", but it also accepts a customized file name as long as it contains '.RData'.
- `intercept.feature` The name of an intercept feature. This is only neccessary if the field data is stored in a file geodatabase and corresponds with the target shapefile containing the field data. If the field data file path does not lead to a geodatabase (a valid shapefile), then leave argument of 'None'.
  - *Note*: File Geodatabases have not yet been tested for this function. If encountering troubles, it may be best to store the target field data as a regular shapefile.
- `newgdb.name` An output geodatabase name. This is only necessary if the 'output.gdb.path' does not lead to a valid geodatabase. A new geodatabase name must contain '.gdb'. If an output geodatabase already exists and is included in the output path, then this parameter can be left at 'None'.

- `quad.sel` Selection of quadpolygons. The default of "None" will include **ALL** 75 100k quadpolygons in the loop. In order to select a valid subset of quadpolygons, this argument must use an object created from `set_extent_manual` in order to properly correspond with the names.
  - *Troubleshooting Tip*: UID of a 100k quadpolygon alone will not work for this parameter, it must contain the 'q' in front of the UID with the '_100blk'. The name of the quadpolygons in 'Covariates' varies slightly from this and may not work correctly yet (bug fixes for this are forthcoming). It may be neccessary to manually alter the character strings output from `select_quads` to properly correspond with 'Covariates' quadpolygons.
- `export` Whether or not to export the merged dbf file. The default 'TRUE' saves the merged file to the output folder file path (note, only the temporary files used to create the merged dbf are deleted at the end of the function).
- `survey` Whether or not the `fielddata.path` leads to field survey data. This is used to determine whether the input data is formatted as a survey or if it is simply a spatial extent used for getting predictions. For field data, leave default 'TRUE', change to 'FALSE' if extracting QuadPolyIDs for a prediciton extent.

Training: Associating Field Data with QuadPolyIDs

Step 1: Install `fsvm` and load required packages

Begin by downloading the latest version of the `fsvm` package from GitLab and loading the necessary packages into your workspace. We will illustrate QuadPolyID extraction for field data using the `fsvm_dummy` data set.

```
#Install latest version of `fsvm`
remotes::install_gitlab("idfg-r/fsvm_package", subdir = "pkg",auth_token = "oYfSyynwxTaobvGua9tF")
> Skipping install of 'fsvm' from a gitlab remote, the SHA1 (3a9f0b61) has not changed since last instal
>   Use `force = TRUE` to force installation

#Load packages
lapply(c("fsvm","rgdal","sp","sf"),require,character.only=T)
> Loading required package: fsvm
> Loading required package: rgdal
> Loading required package: sp
> rgdal: version: 1.5-23, (SVN revision 1121)
> Geospatial Data Abstraction Library extensions to R successfully loaded
> Loaded GDAL runtime: GDAL 3.2.1, released 2020/12/29
> Path to GDAL shared files: C:/Users/rritson/Documents/R/win-library/4.0/rgdal/gdal
> GDAL binary built with GEOS: TRUE
> Loaded PROJ runtime: Rel. 7.2.1, January 1st, 2021, [PJ_VERSION: 721]
> Path to PROJ shared files: C:/Users/rritson/Documents/R/win-library/4.0/rgdal/proj
> PROJ CDN enabled: FALSE
> Linking to sp version:1.4-5
> To mute warnings of possible GDAL/OSR exportToProj4() degradation,
> use options("rgdal_show_exportToProj4_warnings"="none") before loading rgdal.
> Overwritten PROJ_LIB was C:/Users/rritson/Documents/R/win-library/4.0/rgdal/proj
> Loading required package: sf
> Warning: package 'sf' was built under R version 4.0.5
> Linking to GEOS 3.9.0, GDAL 3.2.1, PROJ 7.2.1
> [[1]]
> [1] TRUE
>
> [[2]]
> [1] TRUE
>
> [[3]]
> [1] TRUE
>
> [[4]]
> [1] TRUE
```

Step 2: Select required 100k USGS Quads

In order to efficiently process QuadPolyID extraction, it helps to restrict which 100k USGS quads are looped through in the function. Two functions are included in this package to help with this: `select_quads` and `set_extent_manual`. `select_quads` is used to identify which eCognition polygons intersect with the field data and `set_extent_manual` returns the 100k quads in a format usable in the `quad.sel` parameter of `py_extract_quadpolyID`. `set_extent_manual` can also identify 100k quads corresponding with IDFG Regions, Game Management Units, or 24k quads.

```
## Select QuadPolygons (100k)
#Load data
dummy <- as.data.frame(fsvm::fsvm_dummy)
dummy <- dummy[,1:9]
colnames(dummy) <- c("TranKey","PlotKey","Source","DataType","SampleYear","Easting","Northing","Species

#UIDs of eCognition polygons
quads.uid <- fsvm::select_quads(dat = dummy, x = "Easting", y = "Northing")
> OGR data source with driver: ESRI Shapefile
> Source: "\\hqwildstat\C$\Program Files\R\R-4.0.5\library\fsvm\esri\100kquads", layer: "quad100k_proj"
> with 75 features
> It has 2 fields
> Warning: attribute variables are assumed to be spatially constant throughout all
> geometries
unique(quads.uid)
>  [1] "48116a1" "47116e1" "47115a1" "46116e1" "46115a1" "45115e1" "45116a1"
>  [8] "45115a1" "45114a1" "44114a1" "44113a1" "44112a1" "43114e1" "43113e1"
> [15] "43116a1" "43115a1" "43113a1" "42116e1" "43111a1" "42116a1" "42113a1"

#Set extent (get names of 100k quadpolygons)
quad.sel <- fsvm::set_extent_manual(extent = "UID_100k", selections = quads.uid)
quad.sel
>  [1] "q48116a1_100blk" "q47116e1_100blk" "q47115a1_100blk" "q46116e1_100blk"
>  [5] "q46115a1_100blk" "q45115e1_100blk" "q45116a1_100blk" "q45115a1_100blk"
>  [9] "q45114a1_100blk" "q44114a1_100blk" "q44113a1_100blk" "q44112a1_100blk"
> [13] "q43114e1_100blk" "q43116a1_100blk" "q43115a1_100blk" "q42116e1_100blk"
> [17] "q42116a1_100blk" "q43113e1_100blk" "q43113a1_100blk" "q42113a1_100blk"
> [21] "q43111a1_100blk"

## Additional examples
#IDFG Game management unit
GMU_10A_quads <- fsvm::set_extent_manual(extent = "GMU", selections = "10A")
GMU_10A_quads
> [1] "q47116a1_100blk" "q47115a1_100blk" "q46116e1_100blk" "q46115e1_100blk"
> [5] "q46116a1_100blk" "q46115a1_100blk"

#IDFG Region
```

```
R3_quads <- fsvm::set_extent_manual(extent = "Region", selections = "3")
R3_quads
>  [1] "q46116a1_100blk" "q45116e1_100blk" "q45115e1_100blk" "q45116a1_100blk"
>  [5] "q45115a1_100blk" "q45114e1_100blk" "q45114a1_100blk" "q45113e1_100blk"
>  [9] "q45113a1_100blk" "q44116e1_100blk" "q44115e1_100blk" "q44116a1_100blk"
> [13] "q44115a1_100blk" "q44114e1_100blk" "q44114a1_100blk" "q44117e1_100blk"
> [17] "q44117a1_100blk" "q43116e1_100blk" "q43115e1_100blk" "q43114e1_100blk"
> [21] "q43117e1_100blk" "q43116a1_100blk" "q43115a1_100blk" "q42116e1_100blk"
> [25] "q42115e1_100blk" "q43117a1_100blk" "q42117e1_100blk" "q42116a1_100blk"
> [29] "q42115a1_100blk" "q41116e1_100blk" "q41115e1_100blk" "q42117a1_100blk"
> [33] "q41117e1_100blk" "q43114a1_100blk" "q42114e1_100blk" "q42114a1_100blk"
> [37] "q41114e1_100blk"
```

You can more efficently code your `quad.sel` by nesting `select_quads` within `set_extent_manual`.

```
quad.sel <- fsvm::set_extent_manual(extent = "UID_100k",
            selections = fsvm::select_quads(dat = dummy, x = "Easting", y = "Northing"))
> OGR data source with driver: ESRI Shapefile
> Source: "\\hqwildstat\C$\Program Files\R\R-4.0.5\library\fsvm\esri\100kquads", layer: "quad100k_proj"
> with 75 features
> It has 2 fields
> Warning: attribute variables are assumed to be spatially constant throughout all
> geometries
```

Step 3: Create Shapefile of Field Data

Next you need to create a shapefile from your field data. If your field data is not already in Idaho Transverse Mercator projection, you first need to create a Spatial Points Data Frame using the original projection (WGS84, NAD83, etc.) and then transform it to the Idaho Transverse Mercator projection (the projection of the eCognition polygon). An example of transforming a projection prior to saving is illustrated in the 'Predictions' of this vignette. To look-up your required CRS or proj4string for your projection, visit (https://spatialreference.org/ref/). Since `fsvm_dummy` is already formatted in Idaho Transverse Mercator Projection, we don't need to transform the datum projection before we write the shapefile.

```
#Idaho Transverse Mercator 'proj4string'
idtm <- sp::CRS("+proj=tmerc +lat_0=42 +lon_0=-114 +k=0.9996 +x_0=2500000 +y_0=1200000 +ellps=GRS80 +uni
> Warning in showSRID(uprojargs, format = "PROJ", multiline = "NO", prefer_proj
> = prefer_proj): Discarded datum Unknown based on GRS80 ellipsoid in Proj4
> definition

#Create a Spatial Points Data Frame
fsvm_dummy_spatial <- sp::SpatialPointsDataFrame(data = dummy,
coords = dummy[,c("Easting","Northing")], proj4string = idtm)

#Write Shapefile
rgdal::writeOGR(fsvm_dummy_spatial,
                dsn = "A:/Fine scale vegetation analysis/fsvm_package",
                layer = "fsvm_dummy",
                driver = "ESRI Shapefile", overwrite_layer = T)
> Warning in rgdal::writeOGR(fsvm_dummy_spatial, dsn = "A:/Fine scale vegetation
> analysis/fsvm_package", : Field names abbreviated for ESRI Shapefile driver
```

###Step 4: Extract Quadpoly IDs We are now ready to use the `py_extract_quadpolyID` function for the field data. We are going to use the default path for Python27 and the geodatabase of eCognition polygons (shown above) and leaving the defaults on the optional parameters of intercept.feature and out.RData. Since we have selected the required 100k quads in Step 1, we will specify them here to quad.sel. As the the input data is field survey data, we will specify “TRUE” for the survey parameter.

```
#Store require file paths
py.path <- "C:/Python27/ArcGIS10.6/python.exe"
dummy_shp <- "A:/Fine scale vegetation analysis/fsvm_package/fsvm_dummy.shp"
quad.path <- "A:/Fine scale vegetation analysis/dbases_4modeling/blank_polys100k.gdb"
gdb_path <- "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples"
out_folder <- "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples/Output"

#Execute QuadPolyID Extraction
dummy_quadpolyID <- fsvm::py_extract_quadpolyID(py.path = py.path,
                    fielddata.path = dummy_shp,
                    quad.path = quad.path,
                    output.gdb.path = gdb_path,
                    output.folder.path = out_folder,
                    newgdb.name = "Output.gdb",
                    quad.sel = quad.sel,
                    output.RData = "None", intercept.feature = "None",
                    export = T,
                    survey = T)
> [1] "FieldDataPoints_QuadPolyID.RData will be stored in A:/Fine scale vegetation analysis/fsvm_packag
> [1] "Initializing Python and Loading Function"
> [1] "Reticulating QuadPolyID Extraction..."
> begin script on 2021-07-29 at 09:01:37
> listing selected quadpolygons
> loading target shapefile
> loading feature q48116a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q48116a1_100blk_select_pts exported at 09:12:42
> q48116a1_100blk joined with target shapefile at 09:13:05
> q48116a1_100blk dbf file exported at 09:13:11
> loading feature q47116e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q47116e1_100blk_select_pts exported at 09:14:24
> q47116e1_100blk joined with target shapefile at 09:14:32
> q47116e1_100blk dbf file exported at 09:14:35
> loading feature q47115a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q47115a1_100blk_select_pts exported at 09:15:08
> q47115a1_100blk joined with target shapefile at 09:15:15
> q47115a1_100blk dbf file exported at 09:15:19
> loading feature q46116e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q46116e1_100blk_select_pts exported at 09:15:30
> q46116e1_100blk joined with target shapefile at 09:15:39
> q46116e1_100blk dbf file exported at 09:15:42
> loading feature q46115a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q46115a1_100blk_select_pts exported at 09:15:53
> q46115a1_100blk joined with target shapefile at 09:16:01
> q46115a1_100blk dbf file exported at 09:16:04
> loading feature q45115e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q45115e1_100blk_select_pts exported at 09:16:16
> q45115e1_100blk joined with target shapefile at 09:16:26
> q45115e1_100blk dbf file exported at 09:16:31
> loading feature q45116a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q45116a1_100blk_select_pts exported at 09:16:43
> q45116a1_100blk joined with target shapefile at 09:16:53
> q45116a1_100blk dbf file exported at 09:16:56
> loading feature q45115a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q45115a1_100blk_select_pts exported at 09:18:45
> q45115a1_100blk joined with target shapefile at 09:18:52
> q45115a1_100blk dbf file exported at 09:18:57
> loading feature q45114a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q45114a1_100blk_select_pts exported at 09:19:07
> q45114a1_100blk joined with target shapefile at 09:19:15
> q45114a1_100blk dbf file exported at 09:19:19
> loading feature q44114a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q44114a1_100blk_select_pts exported at 09:19:31
> q44114a1_100blk joined with target shapefile at 09:19:39
> q44114a1_100blk dbf file exported at 09:19:42
> loading feature q44113a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q44113a1_100blk_select_pts exported at 09:20:03
> q44113a1_100blk joined with target shapefile at 09:20:11
> q44113a1_100blk dbf file exported at 09:20:15
> loading feature q44112a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q44112a1_100blk_select_pts exported at 09:22:30
> q44112a1_100blk joined with target shapefile at 09:22:39
> q44112a1_100blk dbf file exported at 09:22:42
> loading feature q43114e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43114e1_100blk_select_pts exported at 09:22:55
> q43114e1_100blk joined with target shapefile at 09:23:07
> q43114e1_100blk dbf file exported at 09:23:10
> loading feature q43116a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43116a1_100blk_select_pts exported at 09:23:25
> q43116a1_100blk joined with target shapefile at 09:23:33
> q43116a1_100blk dbf file exported at 09:23:37
> loading feature q43115a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43115a1_100blk_select_pts exported at 09:23:50
> q43115a1_100blk joined with target shapefile at 09:24:00
> q43115a1_100blk dbf file exported at 09:24:03
> loading feature q42116e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q42116e1_100blk_select_pts exported at 09:25:58
> q42116e1_100blk joined with target shapefile at 09:26:13
> q42116e1_100blk dbf file exported at 09:26:16
> loading feature q42116a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q42116a1_100blk_select_pts exported at 09:26:27
> q42116a1_100blk joined with target shapefile at 09:26:35
> q42116a1_100blk dbf file exported at 09:26:38
> loading feature q43113e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43113e1_100blk_select_pts exported at 09:26:50
> q43113e1_100blk joined with target shapefile at 09:26:59
> q43113e1_100blk dbf file exported at 09:27:04
> loading feature q43113a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43113a1_100blk_select_pts exported at 09:28:00
> q43113a1_100blk joined with target shapefile at 09:28:11
> q43113a1_100blk dbf file exported at 09:28:14
> loading feature q42113a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q42113a1_100blk_select_pts exported at 09:28:43
> q42113a1_100blk joined with target shapefile at 09:28:55
> q42113a1_100blk dbf file exported at 09:28:59
> loading feature q43111a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43111a1_100blk_select_pts exported at 09:29:10
> q43111a1_100blk joined with target shapefile at 09:29:21
> q43111a1_100blk dbf file exported at 09:29:27
> Finished: Output tables are located inA:/Fine scale vegetation analysis/fsvm_package/Vignette_Example
> script complete on 2021-07-29 at 09:29:27
> [1] "Merging output dbfs..."
> Loading required package: foreach
> [1] "Processing: 5 % complete"
> [1] "Processing: 10 % complete"
> [1] "Processing: 14 % complete"
> [1] "Processing: 19 % complete"
> [1] "Processing: 24 % complete"
> [1] "Processing: 29 % complete"
> [1] "Processing: 33 % complete"
> [1] "Processing: 38 % complete"
> [1] "Processing: 43 % complete"
> [1] "Processing: 48 % complete"
> [1] "Processing: 52 % complete"
> [1] "Processing: 57 % complete"
> [1] "Processing: 62 % complete"
> [1] "Processing: 67 % complete"
> [1] "Processing: 71 % complete"
> [1] "Processing: 76 % complete"
> [1] "Processing: 81 % complete"
> [1] "Processing: 86 % complete"
> [1] "Processing: 90 % complete"
> [1] "Processing: 95 % complete"
> [1] "Processing: 100 % complete"
> [1] "Deleting Temporay geodatabse and output files..."
> [1] "Formatting merged output file..."
> [1] "Saving output file to A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples/Output"

head(dummy_quadpolyID)
>                    TranKey                     PlotKey   Source DataTyp SamplYr
> 1 IDFG_2019_q42113c1_3152 IDFG_2019_q42113c1_3152_0.5 IDFG_LPI     LPI    2019
> 2 IDFG_2019_q42113c1_3152   IDFG_2019_q42113c1_3152_1 IDFG_LPI     LPI    2019
> 3 IDFG_2019_q42113c1_3152 IDFG_2019_q42113c1_3152_1.5 IDFG_LPI     LPI    2019
> 4 IDFG_2019_q42113c1_3152   IDFG_2019_q42113c1_3152_2 IDFG_LPI     LPI    2019
> 5 IDFG_2019_q42113c1_3152 IDFG_2019_q42113c1_3152_2.5 IDFG_LPI     LPI    2019
> 6 IDFG_2019_q42113c1_3152   IDFG_2019_q42113c1_3152_3 IDFG_LPI     LPI    2019
>   Easting                 SpecsNm PrcntCv Shape_Leng Shape_Area    QuadPolyID
> 1 2578976   Eriogonum microthecum       0        556       3321 q42113c1_3152
> 2 2578977         Bromus tectorum       0        556       3321 q42113c1_3152
> 3 2578977             Poa secunda       0        556       3321 q42113c1_3152
> 4 2578978 Pseudoroegneria spicata       0        556       3321 q42113c1_3152
> 5 2578978 Pseudoroegneria spicata       0        556       3321 q42113c1_3152
> 6 2578978                    <NA>       0        556       3321 q42113c1_3152
>   Northing     quad
> 1  1239915 q42113c1
> 2  1239916 q42113c1
> 3  1239916 q42113c1
> 4  1239916 q42113c1
> 5  1239916 q42113c1
> 6  1239916 q42113c1
```

We now have field survey data associated with QuadPolyIDs! For specific details on additional data formatting functions or training species distribution models (in a 64-bit session), please see the appropriate vignettes.

Predictions: Associating Extents with QuadPolyIDs

If you are trying to get `fsvm` predictions for a GMU(s), covariates can be queried directly (see `getPredCovs`). Otherwise, you will need to associate your extent shapefile with QuadPolyIDs.

Step 1: Load, Transform, and Save Extent Shapefile (a Homerange)

The spatial extent we will be using is a shapefile of a mule deer VHF home range contour available on the IDFG "KEEP" drive. Once loaded into R, it needs to be projected to Idaho Transverse Mercator in order to extract the QuadPolyIDs from the eCognition polygons.

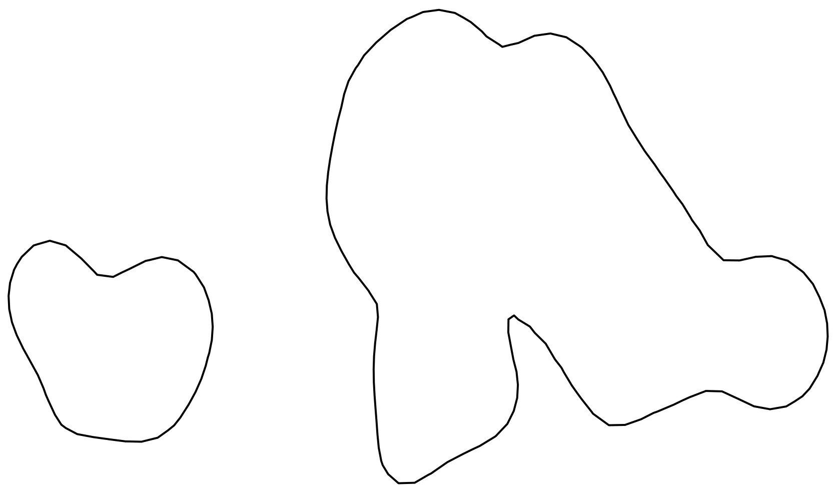
```
#Load shapefile
filepath <- "K:/Wildlife/Wildlife Research/Vegetation Sampling/Vegetation sampling/Summer home ranges/VH
homerange_shp <- rgdal::readOGR(dsn = filepath, layer = "39")
> OGR data source with driver: ESRI Shapefile
> Source: "K:\Wildlife\Wildlife Research\Vegetation Sampling\Vegetation sampling\Summer home ranges\VHF
> with 2 features
> It has 2 fields

#Set projection of original file (WGS84)
sp::proj4string(homerange_shp) <- sp::CRS("+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs")

#Transform to Idaho Transverse Mercator projection
idtm <- sp::CRS("+proj=tmerc +lat_0=42 +lon_0=-114 +k=0.9996 +x_0=2500000 +y_0=1200000 +datum=NAD83 +uni
homerange_shp <- sp::spTransform(homerange_shp, idtm)
```

Next, view and save transformed shapefile

```
#Plot and save homerange kernal
plot(homerange_shp)
```



```
rgdal::writeOGR(homerange_shp,
      dsn = "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples",
      layer = "HR39_idtm",
      driver = "ESRI Shapefile",overwrite_layer = T)
```

After we have saved the re-projected homerange kernel, we are ready to move on to extracting the QuadPolyIDs.

Step 2: Select required 100k USGS Quads

While we had `select_quads` as a helper for identifying quads for our field data, we need to by pass the function and use the internal shapefile to intersect our extent shapefile. Once we have the UIDs of our 100k quads, we can use the `set_extent_manual` function to get our formatted list of selected quads.

```
#Speed up extraction by restricting 100k quads
##Select 100k quads intersecting with homerange
quads<-rgdal::readOGR(paste(system.file('esri',package = 'fsvm',mustWork = T),
                            '100kquads',sep = '/'),layer="quad100k_proj")
> OGR data source with driver: ESRI Shapefile
> Source: "C:\Users\rritson\Documents\R\win-library\4.0\fsvm\esri\100kquads", layer: "quad100k_proj"
> with 75 features
> It has 2 fields
sf.quads <- sf::st_as_sf(quads)
homerange.sf <- sf::st_as_sf(homerange_shp)
X <- sf::st_intersection(homerange.sf,sf.quads)
> Warning: attribute variables are assumed to be spatially constant throughout all
> geometries
uid <- X$UID #use this list in 'fsvm::set_extent_manual'
rm(quads,sf.quads,X,homerange.sf)

quad.sel <- fsvm::set_extent_manual(extent = "UID_100k", selections = uid)
quad.sel
> [1] "q44115a1_100blk" "q43116e1_100blk" "q43115e1_100blk" "q43114e1_100blk"
rm(uid)
```

Now we are ready to execute the Python script to get the QuadPolyIDs within the home range.

Step 3: Extract Quadpoly IDs

The major difference between extracting QuadPolyIDs for field survey data and extents is We need to change the parameter 'survey' to 'FALSE' since the input feature ('fielddata.path', in this case the home range shapefile) is NOT survey data and we are only interested in the particular quad polygons it intersects.

```
#Store required file paths
hr.path<-"A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples/HR39_idtm.shp"
gdb_path <- "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples"
out_folder <- "A:/Fine scale vegetation analysis/fsvm_package/Vignette_Examples/Quads_Output"
#Execute QuadPolyID Extraction
extent.qpid <- fsvm::py_extract_quadpolyID(fielddata.path = hr.path,
                    output.gdb.path = gdb_path,
                    output.folder.path = out_folder,
                    newgdb.name = "OutputExtent.gdb",
                    quad.sel = quad.sel,
                    export = F,
                    survey = F)
> [1] "FieldDataPoints_QuadPolyID.RData will be stored in A:/Fine scale vegetation analysis/fsvm_packag
> [1] "Initializing Python and Loading Function"
> [1] "Reticulating QuadPolyID Extraction..."
> begin script on 2021-07-29 at 09:29:37
> listing selected quadpolygons
> loading target shapefile
> loading feature q44115a1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q44115a1_100blk_select_pts exported at 09:34:41
> q44115a1_100blk joined with target shapefile at 09:36:09
> q44115a1_100blk dbf file exported at 09:36:44
> loading feature q43116e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43116e1_100blk_select_pts exported at 09:37:54
> q43116e1_100blk joined with target shapefile at 09:38:50
> q43116e1_100blk dbf file exported at 09:39:13
> loading feature q43115e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43115e1_100blk_select_pts exported at 09:56:36
> q43115e1_100blk joined with target shapefile at 10:10:00
> q43115e1_100blk dbf file exported at 10:15:05
> loading feature q43114e1_100blk
> selecting ecog polygons that intersect with target shapefile
> exporting
> q43114e1_100blk_select_pts exported at 10:19:07
> q43114e1_100blk joined with target shapefile at 10:20:42
> q43114e1_100blk dbf file exported at 10:21:28
> Finished: Output tables are located inA:/Fine scale vegetation analysis/fsvm_package/Vignette_Example
> script complete on 2021-07-29 at 10:21:28
> [1] "Merging output dbfs..."
> [1] "Processing: 25 % complete"
> [1] "Processing: 50 % complete"
> [1] "Processing: 75 % complete"
> [1] "Processing: 100 % complete"
> [1] "Deleting Temporay geodatabse and output files..."
> [1] "Formatting merged output file..."
> [1] "Extraction Completed"
head(extent.qpid)
>   SP_ID dummy Shape_Leng Shape_Area QuadPolyID     quad
> 1     2     0         56        108 q43114f7_1 q43114f7
> 2     2     0         36         39 q43114f7_2 q43114f7
> 3     2     0        350        500 q43114f7_3 q43114f7
> 4     2     0         80        201 q43114f7_4 q43114f7
> 5     2     0        426        856 q43114f7_5 q43114f7
> 6     2     0        318        652 q43114f7_6 q43114f7
```

You are now ready to use your extent to get species distribution model predictions from the trained models. See the appropriate vignette(s) for further details.