

Возможности R для работы с пространственными данными

Мастер-класс на IV Конференции сообщества природоохраных
ГИС в России
Национальный парк «Валдайский», 05 октября 2019 г.

Никита Платонов (ИПЭЭ РАН)

Обновлено: 2019-10-14 07:41

Планирование (до 27 сентября)

Предлагаемый подход к формированию темы мастер-класса

Планируется, что основные материалы для проведения мастер-класса появятся здесь к 27 сентября. Предполагается, что некоторые дополнения могут быть внесены и в более поздний срок, но с учетом того, чтобы эти изменения не потребовали существенного заимствования времени от занятия. Основной режим проведения мастер-класса - это копирование/вставка фрагмента кода, поэтому этот материал может быть использован дистанционными участниками.

Факультативно задумывается и осваивается вариант занятия с использованием Jupyter R Notebook. Установку необходимых программных компонентов нужно выполнить самостоятельно.

Пожелания по затрагиванию какой-либо темы могут быть приняты
✉ не позднее 27 сентября.

Подготовка (до 05 октября)

Операционные системы пользователя

Не для всех операционных систем есть скомпилированные модули (ядро R, библиотеки). В таком случае модули компилируются, и на это уходит какое-то время. Поэтому если ОС не OS Windows, то этот этап нужно пройти заранее.

Установка базового R

Установить или обновить R, например, отсюда. Актуальная версия 3.6.1. Не ниже версии 3.0.

Установка необходимых библиотек

```
pkgList <- c("rgdal", "sf", "raster", "ggplot2", "leaflet", "mapview", "mapedit"  
    , "knitr", "rmarkdown", "jpeg", "png", "ursa")  
whoisready <- sapply(pkgList, function(pkg) {  
    if (pkg=="ursa") {  
        v <- try(packageVersion("ursa"))  
        if ((inherits(v, "try-error")) || (v<"3.8.15"))  
            install.packages("ursa", repos="http://R-Forge.R-project.org")  
    }  
    if (requireNamespace(pkg))  
        return(TRUE)  
    install.packages(pkg, repos="https://cloud.r-project.org/")  
    requireNamespace(pkg)  
})
```

```
Loading required namespace: rgdal
```

```
Loading required namespace: sf
```

```
Loading required namespace: raster
```

```
Loading required namespace: mapview
```

```
Loading required namespace: mapedit
```

```
Loading required namespace: jpeg
```

Установка необходимых библиотек

```
whoisready
```

rgdal	sf	raster	ggplot2	leaflet	mapview	mapedit
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
knitr	rmarkdown	jpeg	png	ursa		
TRUE	TRUE	TRUE	TRUE	TRUE		

Если отображается TRUE для всех библиотек, то подготовка к занятию осуществлена успешно.

```
c('Everything is ready?'=all(whoisready))
```

```
Everything is ready?  
TRUE
```

Если где-то выскоцило FALSE (например, для библиотеки “foo”), то можно попробовать его установить заново функцией `install.packages("foo")`.

Установка дополнительного программного обеспечения

Pandoc

Pandoc необходим для создания воспроизводимого результата. Этот шаг опциональный, и может быть пропущен, но в этом случае на занятии будет пропущен раздел по публикации результатов.

Ссылка на страницу для скачивания. Для пользователей Windows достаточно перейти к скачиванию актуального релиза, и выбрать либо установщик (*.msi), либо архив (*.zip). Запомнить путь, куда произведена установка и где находится файл pandoc.exe и добавить этот путь в переменную окружения %PATH%, например: WindowsKey+Q, ввести “Переменные среды/Environment Variables”, попасть в окошко “Свойства Системы/System Properties”, нажать на кнопку “Переменные среды/Environment Variables” и отредактировать пользовательскую или системную переменную PATH, добавив путь к pandoc.exe.

Чтобы проверить правильно ли установлен Pandoc, в новой R-сессии:

```
rmarkdown::pandoc_available()
```

```
[1] TRUE
```

Установка дополнительного программного обеспечения

Jupyter R Notebook

Jupyter Notebook для работы с кодом в браузере.

1. Загрузить и установить менеджер Miniconda (проверено на Windows 64bit Python 3.7).
2. Запустить conda (На Windows попробовать WinKey+Q и начать вводить “Anaconda”)
3. Последовательно выполнить команды (в среде Conda, не в среде R):

Команда	Описание
<code>conda install jupyter</code>	установка Jupyter Notebook
<code>conda install -c r r-recommended r-irkernel</code>	Установка R, базовых библиотек и библиотеки для использования R в Jupyter Notebook
<code>conda install -c conda-forge jupyter_text</code>	Установка преобразователя кода R в формат Jupyter Notebook

Предварительно скачайте `main.R` или только код `codeOnly.R`, если будете использовать на занятии Jupyter Notebook. Если проблема с кодировкой остается, попробуйте взять `codeOnly1251.R` Предупреждение: описательный текст (markdown) местами не отформатирован, в частности, не будут отображаться таблицы; также не будут отображаться html-виджеты.

Как использовать материал

Предполагается, что во время занятий будет использоваться интерактивный режим: либо простая среда R (R, запущенный без команд), либо графическая среда R GUI, либо RStudio.

Блок исходного кода выделен моноширинным шрифтом с подсветкой синтаксиса, после которого либо приведен текстовый вывод (более бледный цвет шрифта), либо графический вывод (рисунок, таблица). В среде R нужно вводить код из верхнего блока и сравнивать полученный вывод с содержимым нижнего блока.

К примеру, ниже приведен пример вывода числа π : сверху – команда, снизу – выход.

```
pi  
[1] 3.141593
```

Начало работы

Текущий путь, где мы находимся. Здесь появятся файлы, созданные в процессе занятия.

```
getwd()
```

```
[1] "D:/RAS/2019/SCGIS"
```

Его можно поменять через меню RGui/RStudio или с помощью `setwd()`.

Зафиксируем генератор псевдослучайных чисел на определенную последовательность

```
set.seed(353)
```

```
sample(10)
```

```
[1] 2 7 10 6 4 1 3 9 5 8
```

Начало работы

Команда для проверки кириллицы:

```
print(c('Здесь кириллица?'=="Да!" ), quote=FALSE)
```

```
Здесь кириллица?  
Да!
```

Если вывод не читаемый, то возможные пути решения:

1. Задать кириллицу для символьной локали:

```
if (.Platform$OS.type=="windows")  
  Sys.setlocale("LC_CTYPE", "Russian")
```

2. Если скрипты подключаются через `source()`, то можно настроить запуск R через конфигурационные файлы. Например, скачать `.Rprofile`, разместить в рабочей директории, перегрузить R. Файл имеет следующую структуру:

```
local({  
  options(encoding="UTF-8")  
  Sys.setlocale("LC_CTYPE", "Russian")  
})
```

3. Если скрипт `bar.R` с кириллицей в кодировке UTF-8 запускается из командной строки, то использовать следующие параметры запуска.

```
R --encoding UTF-8 -f bar.R
```

Занятие (05 октября)

R как ГИС?

- R устроен так, что можно реализовать сложные структуры данных, со средствами их инспектирования, что вполне применимо для пространственных данных и их метаданных
- В R есть инструменты для анализа данных
- В R есть инструменты для визуализации
- *Как особенность развития до текущего момента:*
воспроизводимость реализована лучше интерактивности
- В R уже многое сделано по манипуляции с пространственными данными: пользователю не обязательно быть разработчиком

Структуры данных

Числа и имена значений

```
(a1 <- seq(7))
```

```
[1] 1 2 3 4 5 6 7
```

```
str(a1)
```

```
int [1:7] 1 2 3 4 5 6 7
```

```
a2 <- a1
```

```
names(a2) <- format(Sys.Date() + a1 - 1, "%A")
```

```
a2
```

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
1	2	3	4	5	6	7

```
str(a2)
```

```
Named int [1:7] 1 2 3 4 5 6 7  
- attr(*, "names")= chr [1:7] "Monday" "Tuesday" "Wednesday" "Thursday" ...
```

Структуры данных

Целые числа, числа с плавающей точкой

```
str(a2+0L)
```

```
Named int [1:7] 1 2 3 4 5 6 7  
- attr(*, "names")= chr [1:7] "Monday" "Tuesday" "Wednesday" "Thursday" ...
```

```
str(a2+0)
```

```
Named num [1:7] 1 2 3 4 5 6 7  
- attr(*, "names")= chr [1:7] "Monday" "Tuesday" "Wednesday" "Thursday" ...
```

```
typeof(a2+0L)
```

```
[1] "integer"
```

```
typeof(a2+0)
```

```
[1] "double"
```

Структуры данных

Логические значения, строки

```
(a3 <- sample(c(TRUE, FALSE), length(a2), replace=TRUE) )  
[1] FALSE FALSE TRUE TRUE TRUE FALSE TRUE  
  
class(a3)  
[1] "logical"  
  
str(a3)  
logi [1:7] FALSE FALSE TRUE TRUE TRUE FALSE ...  
  
(a4 <- names(a2))  
[1] "Monday"      "Tuesday"      "Wednesday"    "Thursday"     "Friday"       "Saturday"  
[7] "Sunday"  
  
class(a4)  
[1] "character"  
  
str(a4)  
chr [1:7] "Monday" "Tuesday" "Wednesday" "Thursday" "Friday" ...
```

Структуры данных

Списки одинаковой длины

```
(a5 <- list(num=a1[c(1, 3:7)], char=a4[-2]))  
  
$num  
[1] 1 3 4 5 6 7  
  
$char  
[1] "Monday"      "Wednesday"    "Thursday"     "Friday"       "Saturday"    "Sunday"  
  
str(a5)  
  
List of 2  
 $ num : int [1:6] 1 3 4 5 6 7  
 $ char: chr [1:6] "Monday" "Wednesday" "Thursday" "Friday" ...  
  
class(a5)  
  
[1] "list"  
  
length(a5)  
  
[1] 2  
  
sapply(a5, length)  
  
  num  char  
    6      6
```

Структуры данных

Таблицы

```
(a6 <- data.frame(num=a1[c(1, 3:7)], char=a4[-2]))
```

```
  num      char
1   1    Monday
2   3 Wednesday
3   4 Thursday
4   5   Friday
5   6 Saturday
6   7   Sunday
```

```
str(a6)
```

```
'data.frame': 6 obs. of 2 variables:
 $ num : int 1 3 4 5 6 7
 $ char: chr "Monday" "Wednesday" "Thursday" "Friday" ...
```

```
class(a6)
```

```
[1] "data.frame"
```

```
dim(a6)
```

```
[1] 6 2
```

Структуры данных

Списки различной длины

```
(a7 <- list(x=sample(a1, 3), y=sample(a1, 5)) )
```

```
$x  
[1] 7 3 2
```

```
$y  
[1] 4 7 6 5 1
```

```
str(a7)
```

```
List of 2  
$ x: int [1:3] 7 3 2  
$ y: int [1:5] 4 7 6 5 1
```

```
class(a7)
```

```
[1] "list"
```

```
length(a7)
```

```
[1] 2
```

```
sapply(a7, length)
```

```
x y  
3 5
```

Структуры данных

Матрицы

```
(a11 <- matrix(sample(seq(24)), ncol=4, nrow=3))
```

```
 [,1] [,2] [,3] [,4]
[1,]    6   23   11   20
[2,]    2   10   13   21
[3,]    8    1    4    3
```

Размерность массива

```
dim(a11)
```

```
[1] 3 4
```

Структура данных массива

```
str(a11)
```

```
int [1:3, 1:4] 6 2 8 23 10 1 11 13 4 20 ...
```

Структуры данных

Массивы

```
(a8 <- array(sample(24), dim=c(3, 4, 2)))
```

```
, , 1
```

```
 [,1] [,2] [,3] [,4]  
[1,] 15 21 3 16  
[2,] 14 11 23 10  
[3,] 18 24 6 22
```

```
, , 2
```

```
 [,1] [,2] [,3] [,4]  
[1,] 19 9 4 20  
[2,] 1 13 17 5  
[3,] 2 8 7 12
```

```
str(a8)
```

```
int [1:3, 1:4, 1:2] 15 14 18 21 11 24 3 23 6 16 ...
```

```
class(a8)
```

```
[1] "array"
```

Структуры данных

Факторы

```
(a9 <- factor(sample(a4), levels=a4))
```

```
[1] Wednesday Sunday     Friday      Tuesday    Thursday Saturday Monday
Levels: Monday Tuesday Wednesday Thursday Friday Saturday Sunday
```

```
str(a9)
```

```
Factor w/ 7 levels "Monday", "Tuesday", ... : 3 7 5 2 4 6 1
```

```
class(a9)
```

```
[1] "factor"
```

```
(a10 <- factor(sample(a4), levels=a4, ordered=TRUE))
```

```
[1] Thursday Monday      Saturday Tuesday   Wednesday Friday    Sunday
7 Levels: Monday < Tuesday < Wednesday < Thursday < ... < Sunday
```

```
str(a10)
```

```
Ord.factor w/ 7 levels "Monday" < "Tuesday" < ... : 4 1 6 2 3 5 7
```

```
class(a10)
```

```
[1] "ordered" "factor"
```

Визуализация данных

Базовые средства R для визуализации (библиотека `graphics`) пространственных данных:

- `points()` – отображение точек (геометрия POINT)
- `lines()`, `segments()`, `contour()` – отображение линий (геометрия LINESTRING)
- `polygon()`, `polypath()` – отображение полигонов (геометрия POLYGON)
- `image()`, `rasterImage()` – растровые изображения
- `text()`, `legend()` – аннотации
- `axis()`, `mttext()` – рамочное оформление

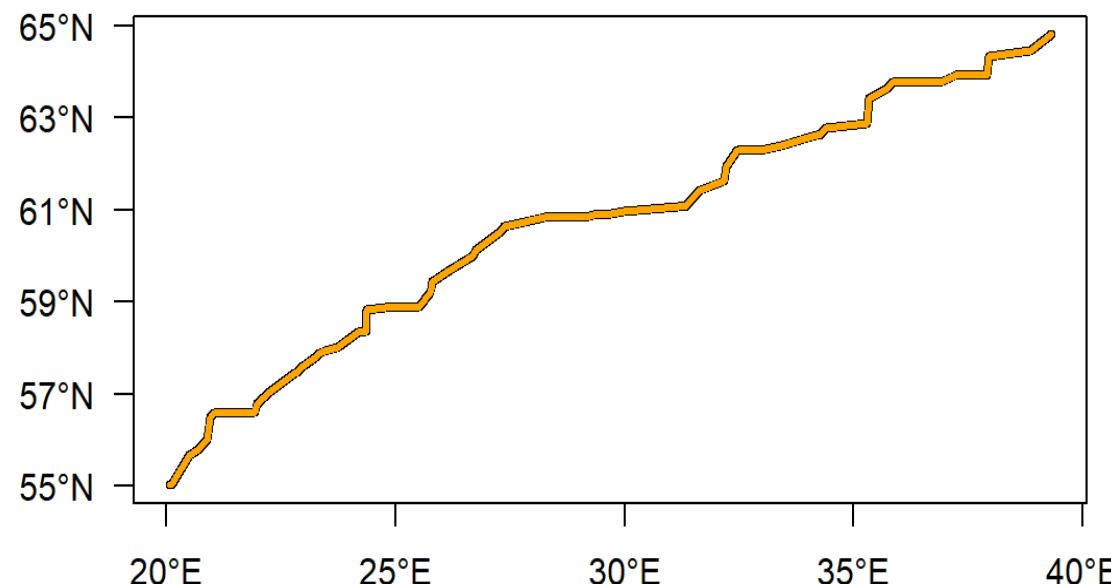
Пример путешествия мимо/через Спб

```
n <- 60
seqx <- seq(20, 40, by=5)
seqy <- seq(55, 65, by=2)
x <- sort(runif(n, min=min(seqx), max=max(seqx)))
y <- sort(runif(n, min=min(seqy), max=max(seqy)))
```

Визуализация данных

Пример путешествия мимо/через Спб

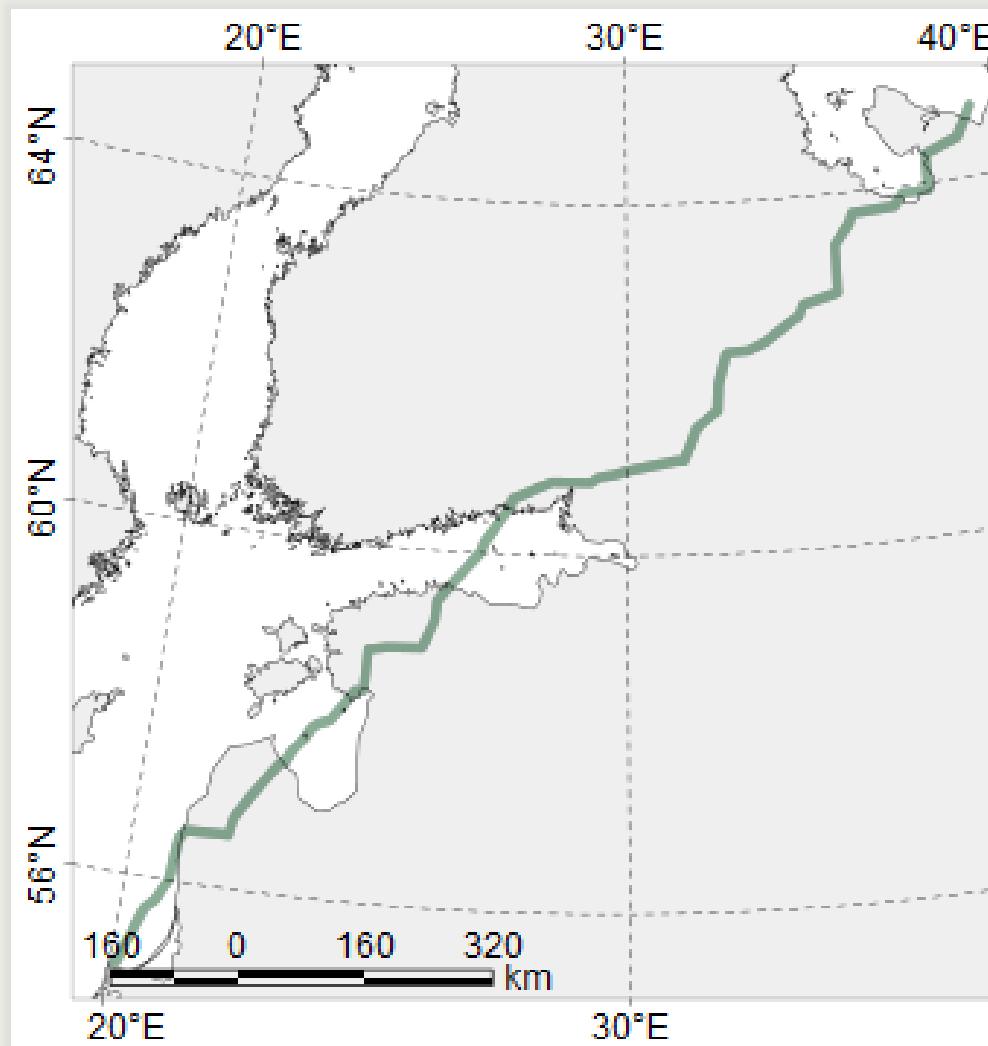
```
plot(x,y,type="n",asp=NA,axes=FALSE,xlab="",ylab="")
lines(x,y,lwd=4,col="black")
lines(x,y,lwd=3,col="orange")
box()
axis(1,at=seqx,lab=paste0(seqx,"°E"),lwd=0,lwd.ticks=1,las=1)
axis(2,at=seqy,lab=paste0(seqy,"°N"),lwd=0,lwd.ticks=1,las=1)
```



Визуализация данных

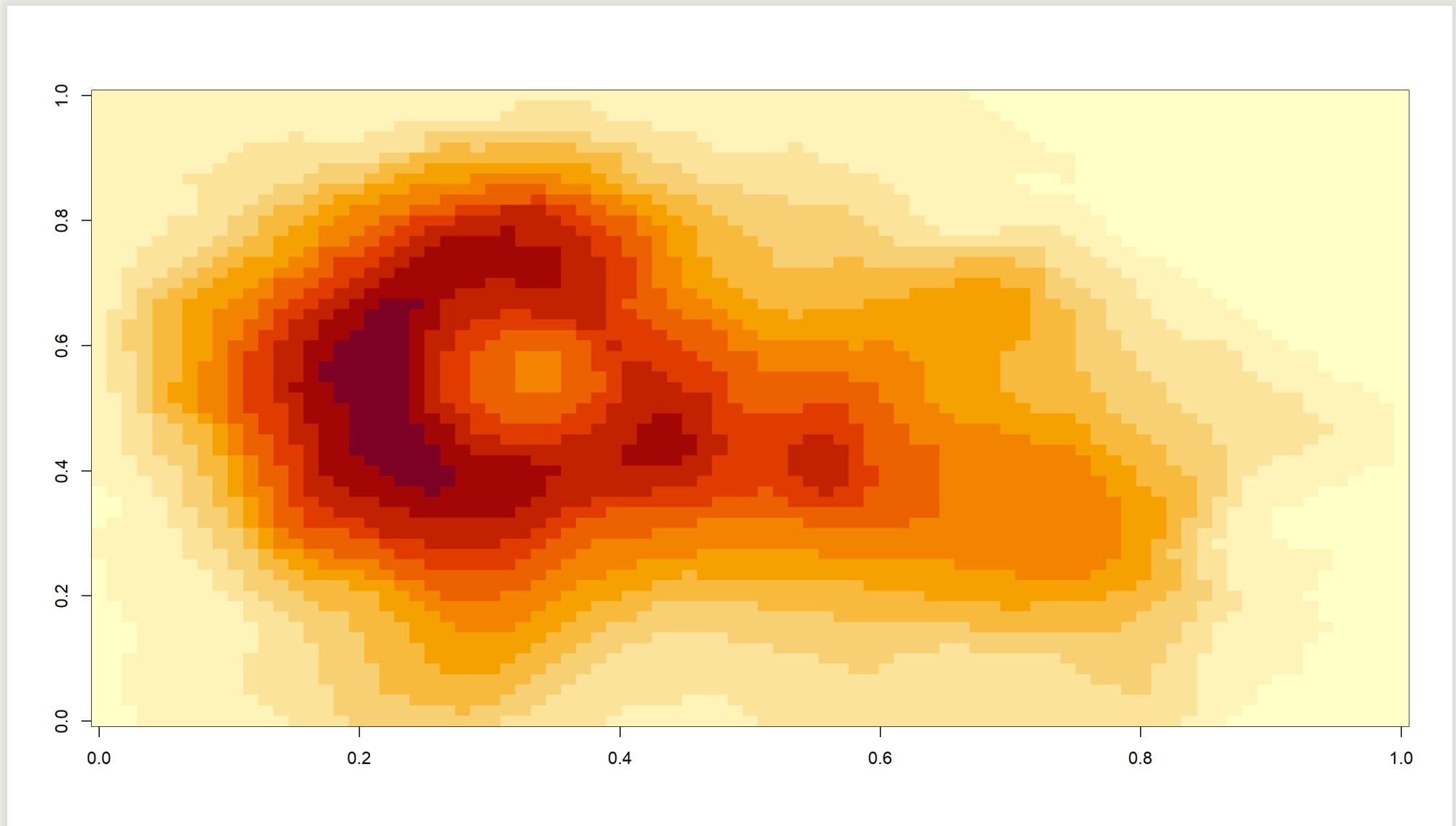
Пример путешествия мимо/через Спб

```
e <- sf:::st_sf(sf:::st_linestring(cbind(x, y)), crs=4326)
ursa::session_grid(NULL)
ursa::glance(e, blank="white", coast.fill="#00000010", height=320, dpi=66
             , col="black", plot.lwd=5)
```



Визуализация данных data(volcano)

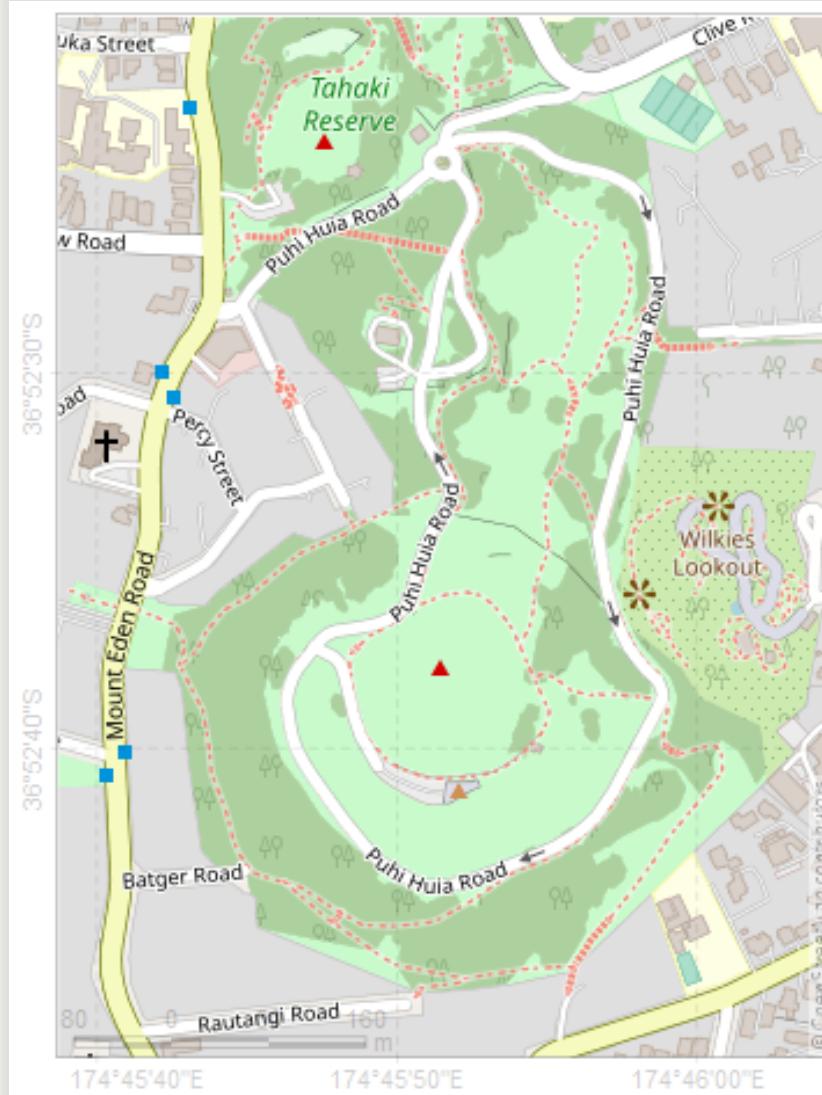
`image(volcano)`



Визуализация данных data(volcano)

Это вулкан Maunga Whau (Mt Eden).

```
try(ursa:::glance("Mount Eden", place="park", dpi=83))
```



Манипуляции с файлами пространственных данных

Векторные данные

Библиотека	Формат	Импорт	Экспорт
sp	GDAL vector drivers	<code>rgdal::readOGR()</code>	<code>rgdal::writeOGR()</code>
sf	GDAL vector drivers	<code>st_read()</code>	<code>st_write()</code>

Растровые данные

Библиотека	Формат	Импорт	Экспорт
sp	GDAL raster drivers	<code>rgdal::readGDAL()</code>	<code>rgdal::writeGDAL()</code>
raster	GDAL raster drivers	<code>raster()</code> , <code>brick()</code> , <code>stack()</code>	<code>writeRaster()</code>
ncdf4	NetCDF	<code>nc_open()</code>	<code>nc_create()</code>
ursa	ENVI	<code>read_envi()</code>	<code>write_envi()</code>

Особенности форматов данных

Растровые данные

- Для хранения многослойные растров используются BSQ/BIL/BIP чередование слоев/строк/пикелей. Самый неэффективный - это BIP. При пространственно-временном анализе можно выбрать BIL, для большинства случаев - BSQ.
- Целочисленный GeoTIFF быстро пишется и читается при использовании функций из библиотеки rgdal

Векторные данные

- Хорошую скорость чтения и записи демонстрирует формат “SQLite” при использовании библиотеки sf.
- “GeoJSON” не очень быстрый.
- При записи использовать опции, предусмотренные для выбранного формата данных
- При записи “ESRI Shapefile” обращать внимания на *.prj, так как у QGIS и ESRI-продуктов разные восприятия файлов проекций.

Особенности форматов данных sf или sp?

- В пользу sf больше аргументов:
 - удобнее, активно развивается, поддерживается
 - в sf объекты класса S3, в sp объекты класса S4 (строже, но медленнее)
 - Эффективность с геометрией POINT выше у sp из-за представления атрибутивной таблицы и геометрии в одной таблице.
 - Ряд библиотек завязаны на формат данных sp, например adehabitatHR.
 - есть sf:::as_Spatial() для преобразования в объекты sp.

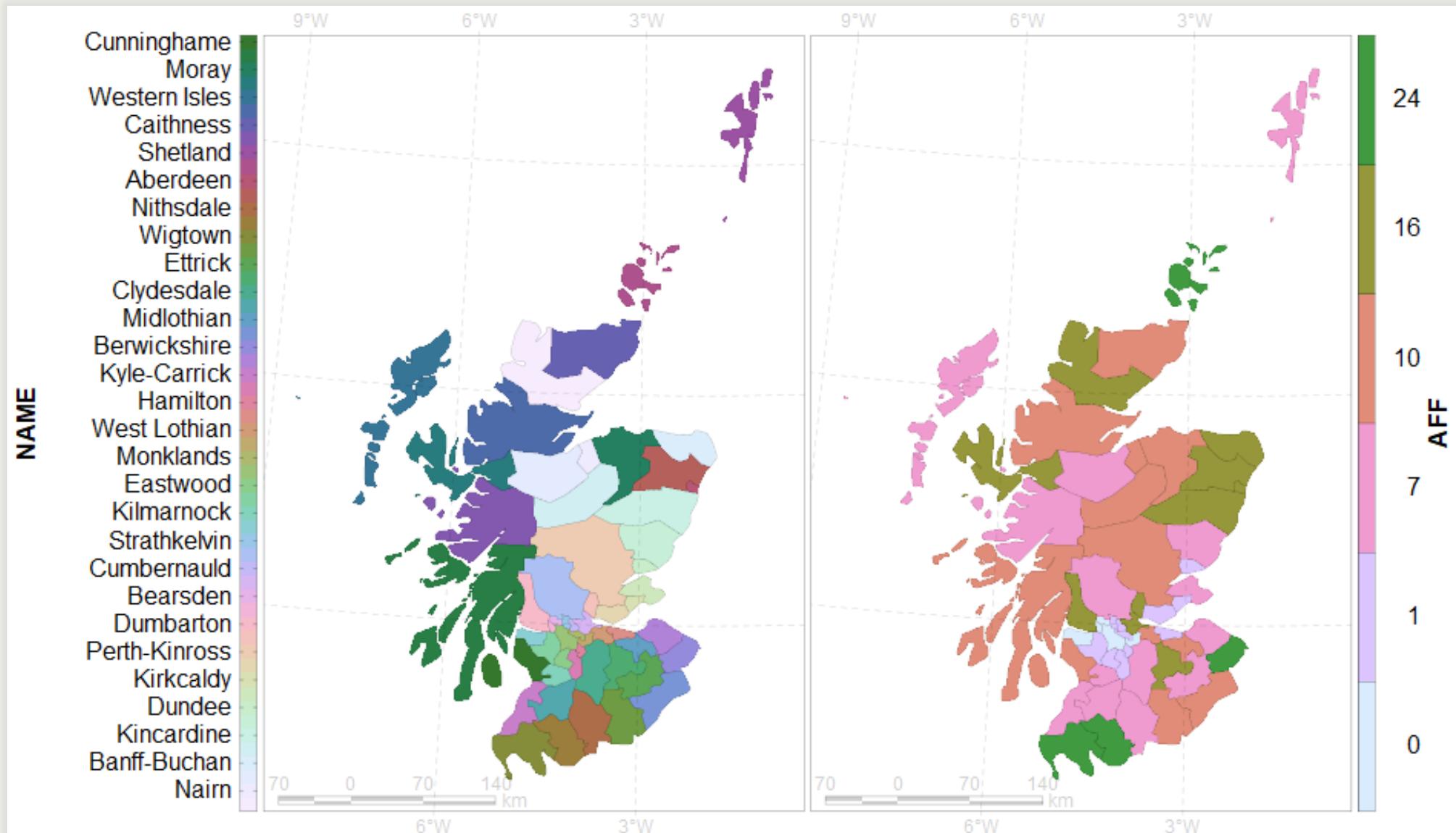
Импорт пространственных данных

Используемые данные для примера

```
(shpname <- system.file("vectors", "scot_BNG.shp", package="rgdal") )  
[1] "C:/Software/Rtools/library/rgdal/vectors/scot_BNG.shp"  
file.exists(shpname)  
[1] TRUE  
(tifname <- system.file("pictures/cea.tif", package="rgdal") )  
[1] "C:/Software/Rtools/library/rgdal/pictures/cea.tif"  
file.exists(tifname)  
[1] TRUE
```

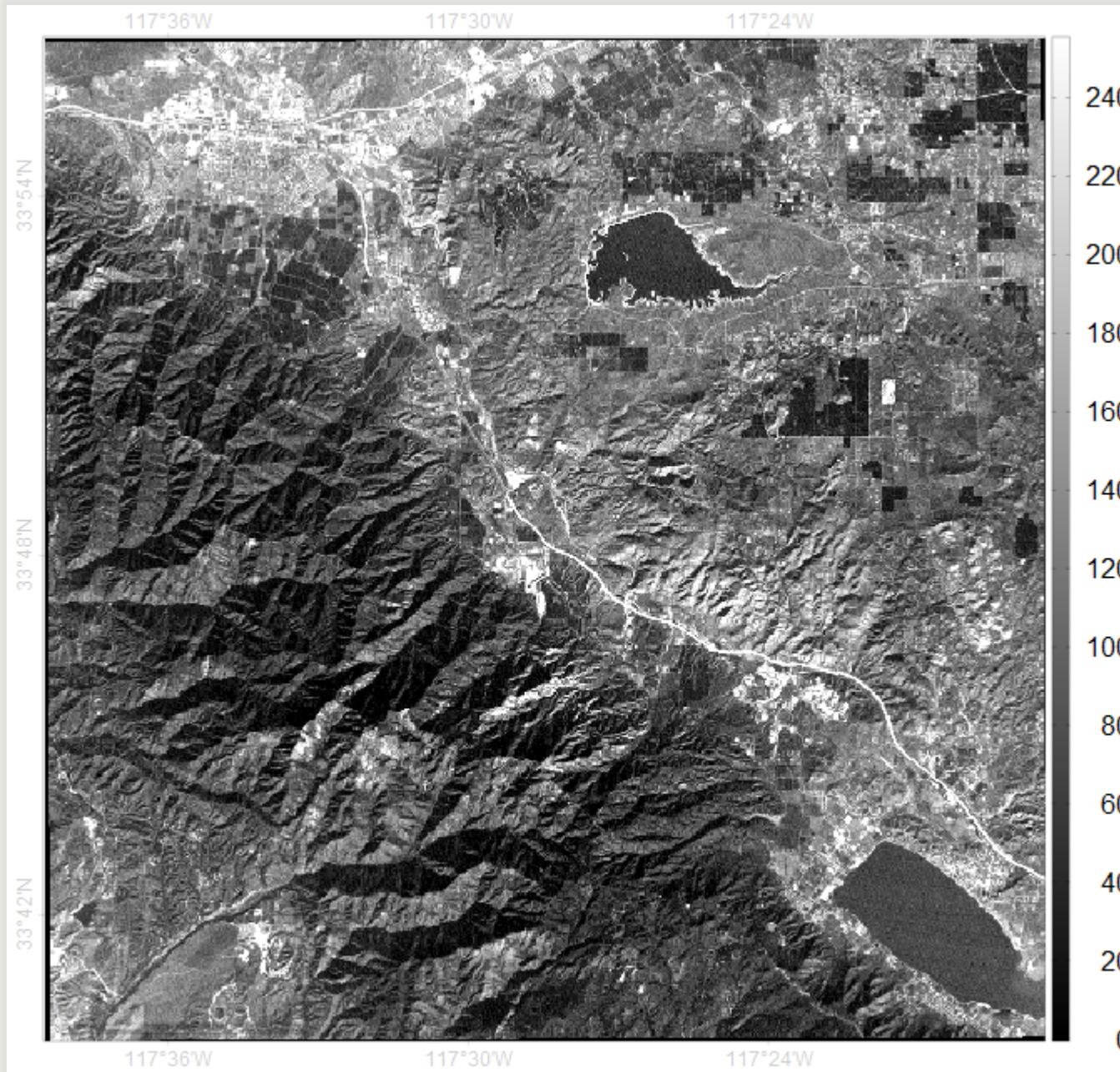
Импорт пространственных данных

```
ursa::session_grid(NULL)
ursa::glance(shpname, coast=FALSE, field=" (NAME | AFF)", blank="white"
             , legend=list("left", "right"), dpi=88)
```



Импорт пространственных данных

```
ursa::session_grid(NULL)  
ursa::glance(tifname, coast=FALSE, pal.from=0, pal=c("black", "white"), dpi=96)
```



Импорт пространственных данных

Векторные данные – rgdal/sp

```
rgdal::ogrInfo(shpname)
```

```
Source: "C:\Software\Rtools\library\rgdal\vectors\scot_BNG.shp", layer: "scot_BNG"
Driver: ESRI Shapefile; number of rows: 56
Feature type: wkbPolygon with 2 dimensions
Extent: (7094.552 529495) - (468285.5 1218342)
CRS: +proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000
LDID: 0
Number of fields: 13
      name    type length typeName
1   SP_ID     4      5   String
2   NAME     4     13   String
3   ID_X     2     19   Real
4   COUNT     2     19   Real
5   SMR      2     19   Real
6   LONG     2     19   Real
7   LAT      2     19   Real
8   PY       2     19   Real
9   EXP      2     19   Real
10  AFF      2     19   Real
11  X_COOR   2     19   Real
12  Y_COOR   2     19   Real
```

```
b.sp <- rgdal::readOGR(shpname)
```

```
OGR data source with driver: ESRI Shapefile
Source: "C:\Software\Rtools\library\rgdal\vectors\scot_BNG.shp", layer: "scot_BNG"
with 56 features
It has 13 fields
```

Импорт пространственных данных

Векторные данные – rgdal/sp

```
str(head(b.sp, 2))
```

```
Formal class 'SpatialPolygonsDataFrame' [package "sp"] with 5 slots
..@ data      :'data.frame': 2 obs. of 13 variables:
.. ..$ SP_ID   : chr [1:2] "0" "1"
.. ..$ NAME    : chr [1:2] "Sutherland" "Nairn"
.. ..$ ID_x    : num [1:2] 12 13
.. ..$ COUNT   : num [1:2] 5 3
.. ..$ SMR     : num [1:2] 279 278
.. ..$ LONG    : num [1:2] 58.1 57.5
.. ..$ LAT     : num [1:2] 4.64 3.98
.. ..$ PY      : num [1:2] 37521 29374
.. ..$ EXP     : num [1:2] 1.8 1.1
.. ..$ AFF     : num [1:2] 16 10
.. ..$ X_COOR  : num [1:2] 247757 289592
.. ..$ Y_COOR  : num [1:2] 924954 842305
.. ..$ ID_y    : num [1:2] 12 13
..@ polygons  :List of 2
.. ..$ :Formal class 'Polygons' [package "sp"] with 5 slots
.. ... .@ Polygons :List of 1
```

iss4 (b.sp)

[1] TRUE

slotNames (b.sp)

```
[1] "data"           "polygons"       "plotOrder"      "bbox"          "proj4string"
```

Импорт пространственных данных

Векторные данные – sf

```
b.sf <- sf::st_read(shpname)
```

```
Reading layer `scot_BNG' from data source `C:\Software\Rtools\library\rgdal\ve
Simple feature collection with 56 features and 13 fields
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 7094.552  ymin: 529495  xmax: 468285.5  ymax: 1218342
epsg (SRID):   NA
proj4string:    +proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y
```

Импорт пространственных данных

Векторные данные – sf

```
str(b.sf)
```

```
Classes 'sf' and 'data.frame': 56 obs. of 14 variables:
$ SP_ID    : chr  "0" "1" "2" "3" ...
$ NAME     : chr  "Sutherland" "Nairn" "Inverness" "Banff-Buchan" ...
$ ID_x     : num  12 13 19 2 17 16 21 50 15 25 ...
$ COUNT    : num  5 3 9 39 2 9 16 6 17 19 ...
$ SMR      : num  279 278 163 450 187 ...
$ LONG     : num  58.1 57.5 57.2 57.6 57.1 ...
$ LAT      : num  4.64 3.98 4.73 2.36 4.09 3 2.98 3.2 3.1 3.3 ...
$ PY       : num  37521 29374 162867 231337 27075 ...
$ EXP      : num  1.8 1.1 5.5 8.7 1.1 4.6 10.5 19.6 7.8 15.5 ...
$ AFF      : num  16 10 7 16 10 16 7 1 7 1 ...
$ X_COOR   : num  247757 289592 249685 385776 280492 ...
$ Y_COOR   : num  924954 842305 825855 852378 801036 ...
$ ID_y     : num  12 13 19 2 17 16 21 50 15 25 ...
$ geometry:sfc MULTIPOLYGON of length 56; first list element: List of 1
..$ :List of 1
.. ..$ : num [1:73, 1:2] 254218 254359 252991 244974 259133 ...
.. - attr(*, "class")= chr "XY" "MULTIPOLYGON" "sfg"
- attr(*, "sf_column")= chr "geometry"
- attr(*, "agr")= Factor w/ 3 levels "constant", "aggregate", ...: NA NA NA NA N
.. - attr(*, "names")= chr "SP_ID" "NAME" "ID_x" "COUNT" ...
```

Импорт пространственных данных

Растровые данные – rgdal/sp

Получение данных напрямую

```
d1 <- rgdal::readGDAL(tifname)
```

```
C:/Software/Rtools/library/rgdal/pictures/cea.tif has GDAL driver GTiff  
and has 515 rows and 514 columns
```

```
str(d1)
```

```
Formal class 'SpatialGridDataFrame' [package "sp"] with 4 slots  
..@ data      :'data.frame': 264710 obs. of 1 variable:  
.. ..$ band1: int [1:264710] 0 0 0 0 0 0 0 0 0 ...  
..@ grid      :Formal class 'GridTopology' [package "sp"] with 3 slots  
.. .. ..@ cellcentre.offset: Named num [1:2] -28463 4225003  
.. .. ..- attr(*, "names")= chr [1:2] "x" "y"  
.. .. ..@ cellsize       : num [1:2] 60 60  
.. .. ..@ cells.dim     : int [1:2] 514 515  
..@ bbox        : num [1:2, 1:2] -28493 4224973 2358 4255885  
.. ..- attr(*, "dimnames")=List of 2  
.. .. ..$ : chr [1:2] "x" "y"  
.. .. ..$ : chr [1:2] "min" "max"  
..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slot  
.. .. ..@ projargs: chr "+proj=cea +lon_0=-117.33333333333 +lat_ts=33.75 +x_0=0 +y_
```

```
summary(d1@data[[1]])
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.0	58.0	99.0	103.1	140.0	255.0

Импорт пространственных данных

Растровые данные – rgdal/sp

Получение данных более гибким способом

```
md <- rgdal:::GDALinfo(tifname)
```

```
Warning in rgdal:::GDALinfo(tifname): statistics not supported by this
driver
```

```
mdname <- names(md)
attributes(md) <- NULL
names(md) <- mdname
md
```

rows	columns	bands	ll.x	ll.y
515.00000	514.00000	1.00000	-28493.16678	4224973.14326
res.x	res.y	oblique.x	oblique.y	
60.02214	60.02214	0.00000	0.00000	

```
dset <- methods::new("GDALReadOnlyDataset",tifname)
d2 <- rgdal:::getRasterData(dset, offset=c(0,0), region.dim=md[c("rows","columns")])
str(d2)
```

```
int [1:514, 1:515] 0 0 0 0 0 0 0 0 0 0 ...
```

```
summary(c(d2))
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.0	58.0	99.0	103.1	140.0	255.0

Импорт пространственных данных

Растровые данные – raster

```
(d3 <- raster::brick(tifname))

class      : RasterBrick
dimensions : 515, 514, 264710, 1 (nrow, ncol, ncell, nlayers)
resolution : 60.02214, 60.02214 (x, y)
extent     : -28493.17, 2358.212, 4224973, 4255885 (xmin, xmax, ymin, ymax)
crs        : +proj=cea +lon_0=-117.33333333333 +lat_ts=33.75 +x_0=0 +y_0=0 +datum=NAD27
source     : C:/Software/Rtools/library/rgdal/pictures/cea.tif
names      : cea
min values : 0
max values : 255

v3 <- d3[] ## 'd3[]' то же, что и 'raster::getValues(d3)'
c(d3=object.size(d3), v3=object.size(v3))

d3          v3
8256 1059160

str(v3)

int [1:264710, 1] 0 0 0 0 0 0 0 0 0 ...
- attr(*, "dimnames")=List of 2
..$ : NULL
..$ : chr "cea"

summary(c(v3))

Min. 1st Qu. Median   Mean 3rd Qu.   Max.
0.0    58.0   99.0   103.1  140.0   255.0
```

Характеристики пространственных данных

Атрибутивная таблица

sp:

```
head(slot(b.sp, "data"))
```

Характеристики пространственных данных

Атрибутивная таблица

sf:

```
head(sf::st_set_geometry(b.sf, NULL))
```

Характеристики пространственных данных

Геометрия

sp:

```
g.sp <- sp::geometry(b.sp)  
g.sp ## Не показано: много строк  
str(head(g.sp, 2))
```

```
Formal class 'SpatialPolygons' [package "sp"] with 4 slots  
..@ polygons :List of 2  
.. ..$ :Formal class 'Polygons' [package "sp"] with 5 slots  
.. .. .. @ Polygons :List of 1  
.. .. .. ..$ :Formal class 'Polygon' [package "sp"] with 5 slots  
.. .. .. .. ..@ labpt : num [1:2] 247754 924885  
.. .. .. .. ..@ area : num 3.92e+09  
.. .. .. .. ..@ hole : logi FALSE  
.. .. .. .. ..@ ringDir: int 1  
.. .. .. .. ..@ coords : num [1:73, 1:2] 254218 254359 252991 244974 2  
.. .. .. .. ..@ plotOrder: int 1  
.. .. .. ..@ labpt : num [1:2] 247754 924885  
.. .. .. ..@ ID : chr "0"  
.. .. .. ..@ area : num 3.92e+09  
.. ..$ :Formal class 'Polygons' [package "sp"] with 5 slots  
.. .. .. ..@ Polygons :List of 1  
.. .. .. .. ..$ :Formal class 'Polygon' [package "sp"] with 5 slots  
.. .. .. .. ..@ labpt : num [1:21 200500 042225
```

Характеристики пространственных данных

Геометрия

sf:

```
(head(g.sf <- sf::st_geometry(b.sf), 2))
```

```
Geometry set for 2 features
geometry type:  MULTIPOLYGON
dimension:      XY
bbox:           xmin: 204669.7 ymin: 825310 xmax: 306671.2 ymax: 974566.2
epsg (SRID):   NA
proj4string:    +proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=0
```

```
MULTIPOLYGON (((254218.3 967027, 254358.6 96696...
```

```
MULTIPOLYGON (((292542.2 859596.3, 301398.8 836...
```

```
str(g.sf)
```

```
sfc_MULTIPOLYGON of length 56; first list element: List of 1
$ :List of 1
..$ : num [1:73, 1:2] 254218 254359 252991 244974 259133 ...
- attr(*, "class")= chr [1:3] "XY" "MULTIPOLYGON" "sfg"
```

Характеристики пространственных данных

Проекция

В R данные проекции представляются в виде PROJ.4, поэтому при импорте/экспорте данных осуществляется двойное WKT -> PROJ4 -> WKT преобразование

sp:

```
sp::proj4string(b.sp)
[1] "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000
```

sf:

```
sf::st_crs(b.sf)
Coordinate Reference System:
  No EPSG code
  proj4string: "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.9996012717 +x_0=400000 +y_0=-100000"
```

Характеристики пространственных данных

Пространственный охват

sp:

```
sp::bbox(b.sp)
```

	min	max
x	7094.552	468285.5
y	529495.039	1218342.5

sf:

```
sf::st_bbox(b.sf)
```

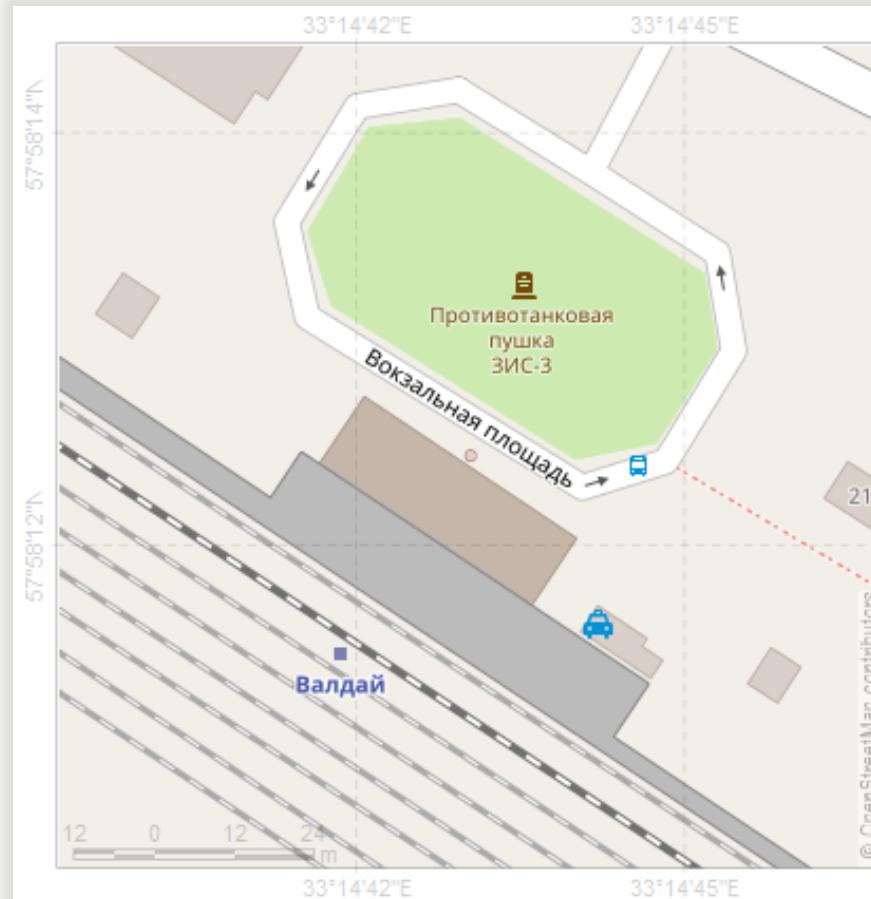
xmin	ymin	xmax	ymax
7094.552	529495.039	468285.495	1218342.493

Создание пространственных данных

Выйдем из здания вокзала ст. Валдай и создадим точечный Spatial-объект:

```
pt0 <- data.frame(lon=33.24529, lat=57.97012)
sp::coordinates(pt0) <- ~lon+lat
sp::proj4string(pt0) <- sp::CRS("+init=epsg:4326")

ursa::glance(pt0, style="mapnik", basemap.order="before", basemap.alpha=1, dpi=91)
```



Создание пространственных данных

Смотрим на структуру данных:

```
str(pt0)
```

```
Formal class 'SpatialPoints' [package "sp"] with 3 slots
..@ coords      : num [1, 1:2] 33.2 58
... ..- attr(*, "dimnames")=List of 2
...   ..$ : chr "1"
...   ..$ : chr [1:2] "lon" "lat"
..@ bbox        : num [1:2, 1:2] 33.2 58 33.2 58
... ..- attr(*, "dimnames")=List of 2
...   ..$ : chr [1:2] "lon" "lat"
...   ..$ : chr [1:2] "min" "max"
..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slot
... ..@ projargs: chr "+init=epsg:4326 +proj=longlat +datum=WGS84 +no_defs
```

Изменим проекцию для расчета расстояний по координатам:

```
pt0 <- sp::spTransform(pt0, "+proj=laea +lat_0=58 +lon_0=35 +datum=WGS84")
```

Извлечем координаты:

```
xy <- sp::coordinates(pt0)
```

Создание пространственных данных

Решаем перемещаться минутными сегментами в течение часа:

```
n <- 60
```

На основе координат создаем таблицу перемещения. По умолчанию весь период стоим на месте и смотрим на север:

```
loc <- data.frame(step=seq(0,n), look=pi/2, x=xy[,1], y=xy[,2])
```

Скорость перемещений (в м мин⁻¹) в течение минуты случайна:

```
segment <- runif(n,min=5,max=80)
```

```
str(segment)
```

```
num [1:60] 33.97 13.93 23.8 9.04 21.29 ...
```

Задаем, что если следующий сегмент длинный, то отклонение направления от предыдущего сегмента будет меньше:

```
angle <- sapply(1-segment/100, function(x) runif(1,min=-x*pi,max=x*pi))
```

```
str(angle)
```

```
num [1:60] 0.978 2.263 0.686 -2.839 2.318 ...
```

Создание пространственных данных

Заполняем последующие шаги на основе предыдущих:

```
for (i in seq(n)) {  
  loc$look[i+1] <- (loc$look[i]+angle[i]) %% (2*pi)  
  loc$x[i+1] <- loc$x[i]+segment[i]*cos(loc$look[i+1])  
  loc$y[i+1] <- loc$y[i]+segment[i]*sin(loc$look[i+1])  
}  
head(loc)
```

	step	look	x	y
1	0	1.570796	-103834.6	-1982.122
2	1	2.548995	-103862.8	-1963.148
3	2	4.811903	-103861.4	-1977.006
4	3	5.497473	-103844.6	-1993.837
5	4	2.657984	-103852.6	-1989.634
6	5	4.976429	-103847.0	-2010.182

Создание пространственных данных

Создадим линейный sf-объект

```
tr <- vector("list", n)
```

Матрица (с двумя столбцами) координат задается как LINESTRING:

```
for (i in seq(n))
  tr[[i]] <-
    sf::st_linestring(matrix(c(loc$x[i], loc$y[i], loc$x[i+1], loc$y[i+1])
      , ncol=2, byrow=TRUE))
str(tr)
```

```
List of 60
$ : 'XY' num [1:2, 1:2] -103835 -103863 -1982 -1963
$ : 'XY' num [1:2, 1:2] -103863 -103861 -1963 -1977
$ : 'XY' num [1:2, 1:2] -103861 -103845 -1977 -1994
$ : 'XY' num [1:2, 1:2] -103845 -103853 -1994 -1990
$ : 'XY' num [1:2, 1:2] -103853 -103847 -1990 -2010
$ : 'XY' num [1:2, 1:2] -103847 -103808 -2010 -2067
$ : 'XY' num [1:2, 1:2] -103808 -103748 -2067 -2074
$ : 'XY' num [1:2, 1:2] -103748 -103733 -2074 -2098
$ : 'XY' num [1:2, 1:2] -103733 -103769 -2098 -2129
$ : 'XY' num [1:2, 1:2] -103769 -103803 -2129 -2149
$ : 'XY' num [1:2, 1:2] -103803 -103797 -2149 -2154
$ : 'XY' num [1:2, 1:2] -103797 -103737 -2154 -2160
$ : 'XY' num [1:2, 1:2] -103737 -103700 -2160 -2164
$ : 'XY' num [1:2, 1:2] -103700 -103696 -2164 -2116
$ : 'XY' num [1:2, 1:2] -103696 -103701 -2116 -2120
$ : 'XY' num [1:2, 1:2] 103701 103710 2120 2101
```

Создание пространственных данных

Набор сегментов объединяем в геометрию:

```
tr <- sf:::st_sf(tr, crs=sp::proj4string(pt0))  
str(tr)  
  
sfc_LINESTRING of length 60; first list element: 'XY' num [1:2, 1:2] -103835
```

С геометрией связываем атрибутивную таблицу.

```
tr <- sf:::st_sf(step=seq(n), segment=segment, geometry=tr)  
str(tr)  
  
Classes 'sf' and 'data.frame': 60 obs. of 3 variables:  
 $ step      : int 1 2 3 4 5 6 7 8 9 10 ...  
 $ segment   : num 33.97 13.93 23.8 9.04 21.29 ...  
 $ geometry:sfc_LINESTRING of length 60; first list element: 'XY' num [1:2, 1:  
 - attr(*, "sf_column")= chr "geometry"  
 - attr(*, "agr")= Factor w/ 3 levels "constant", "aggregate", ...: NA NA  
 ..- attr(*, "names")= chr "step" "segment"
```

Статическая визуализация

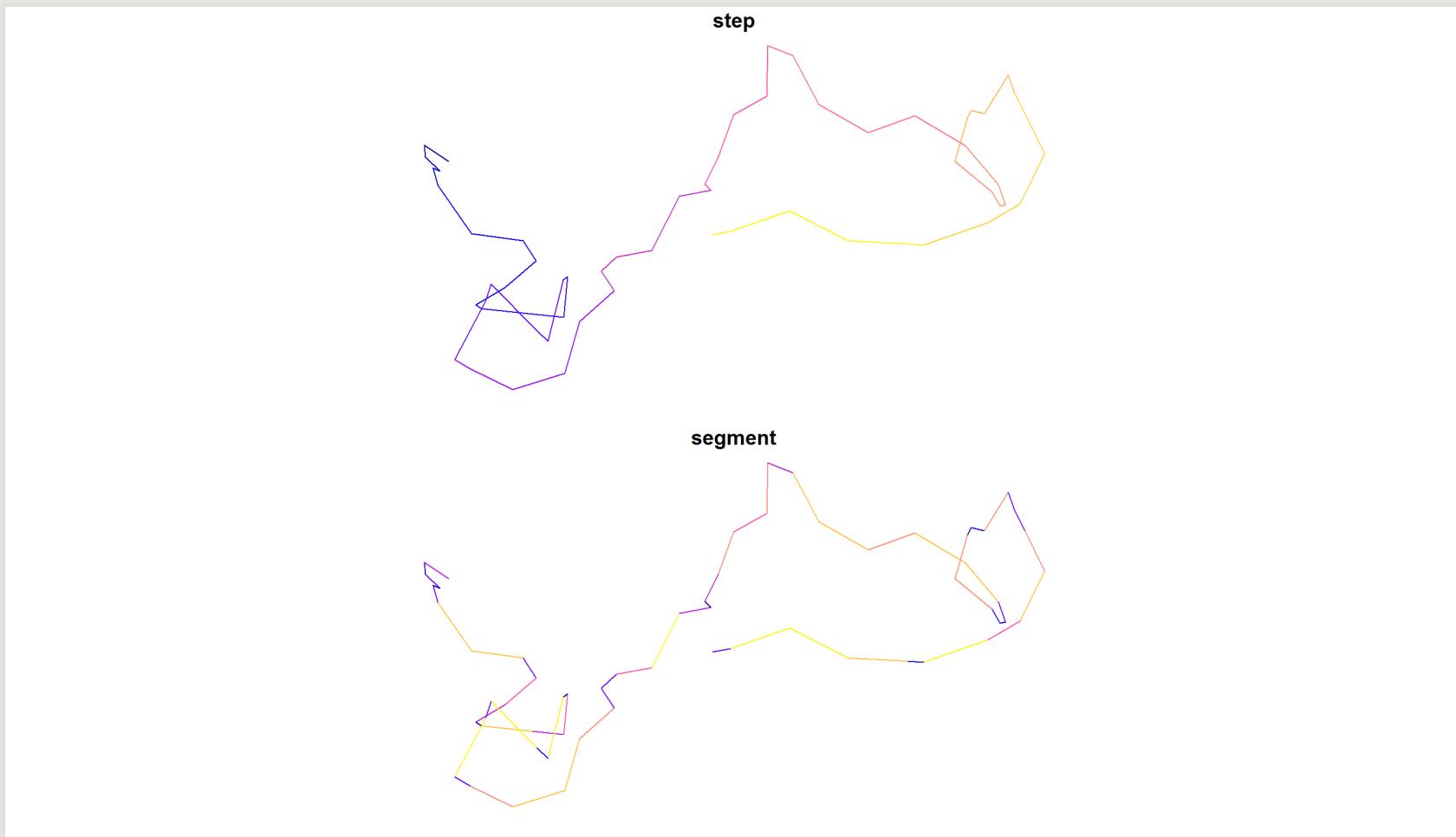
```
ursa::session_grid(NULL)
ursa::glance(tr, style="mapnik"
             , layout=c(1, 2), legend=list("left", list("bottom", 2))), las=1, dpi=96)
```



Статическая визуализация

Графическое отображение переопределенной под класс объекта функцией `plot()` средствами библиотеки `sp`^(факультативно) и `sf` развито слабо...

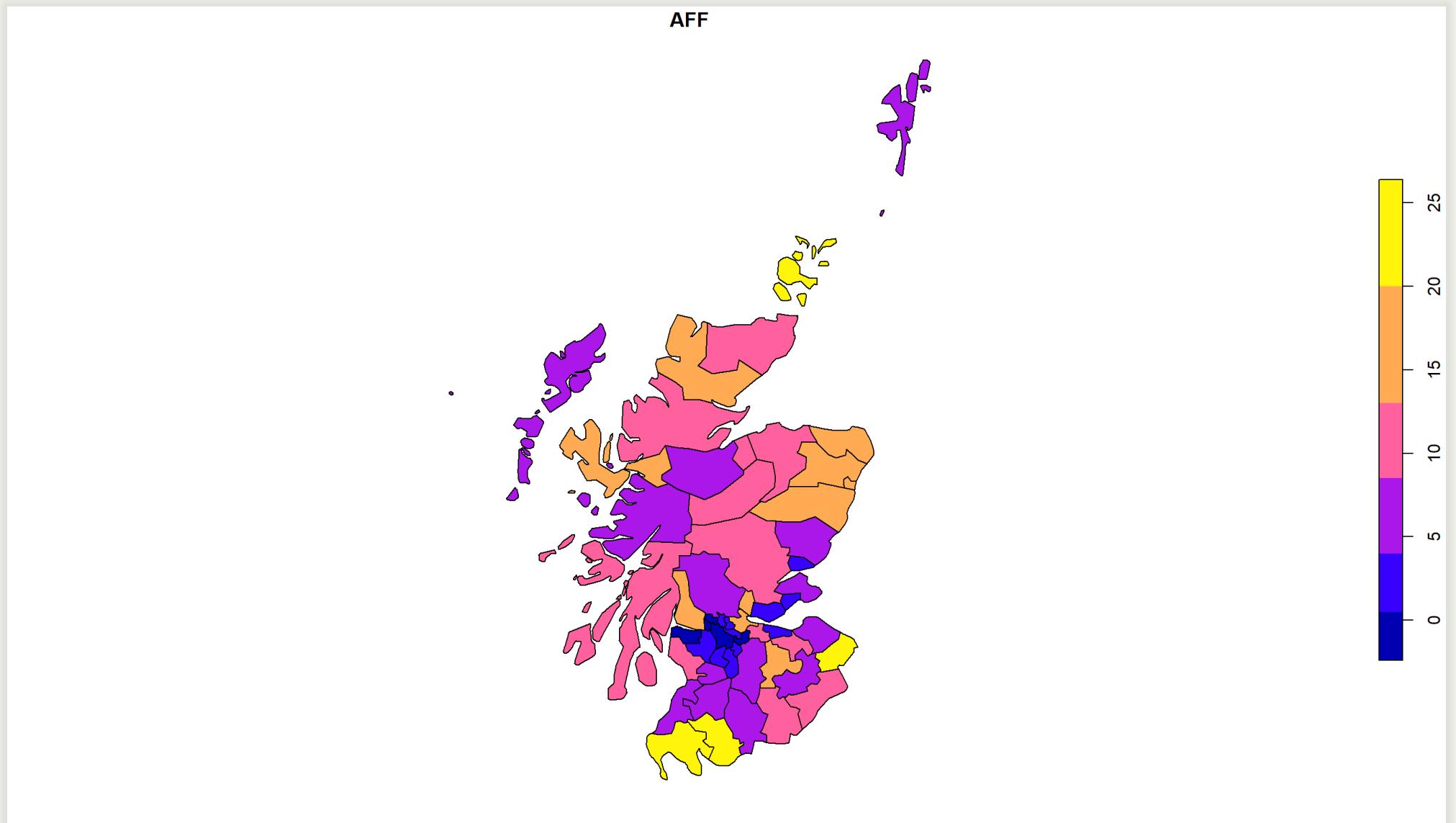
```
plot(tr)
```



Статическая визуализация

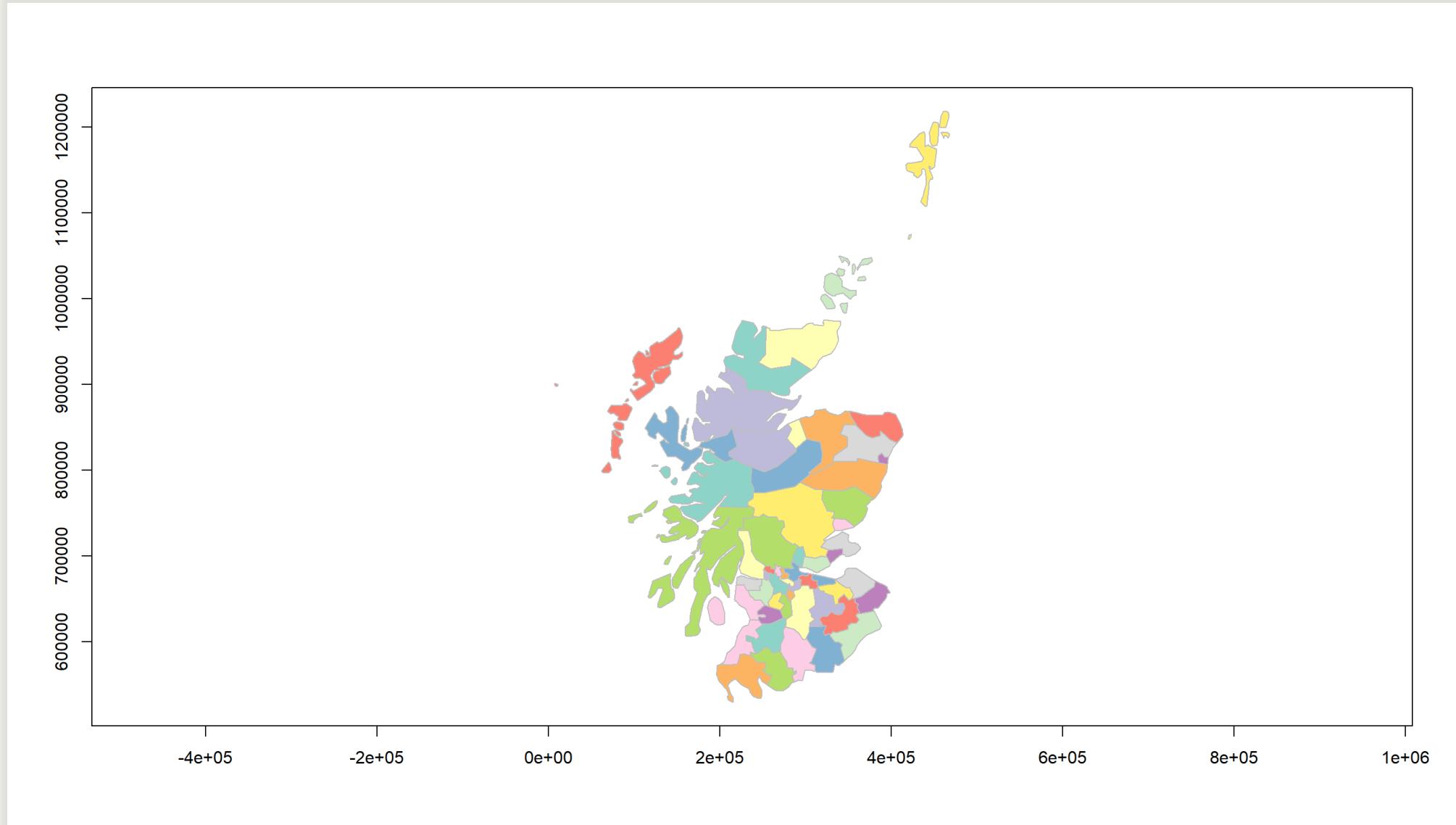
Возвращаясь к ранее загруженным данным:

```
plot(b.sf["AFF"])
```



Статическая визуализация

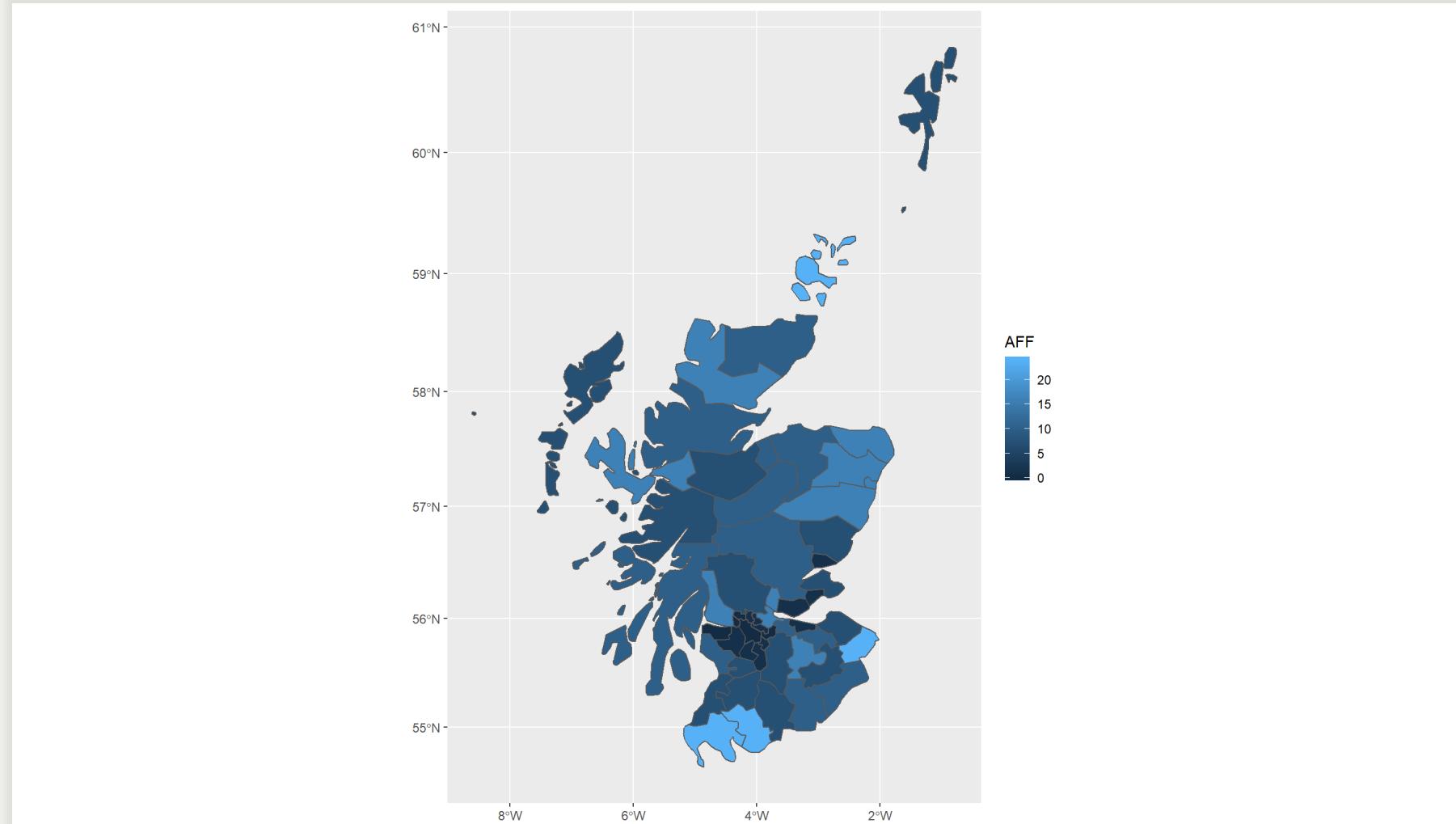
```
plot(sf::st_geometry(b.sf), col=sf::sf.colors(12, categorical=TRUE),  
     border='grey', axes=TRUE)
```



Статическая визуализация

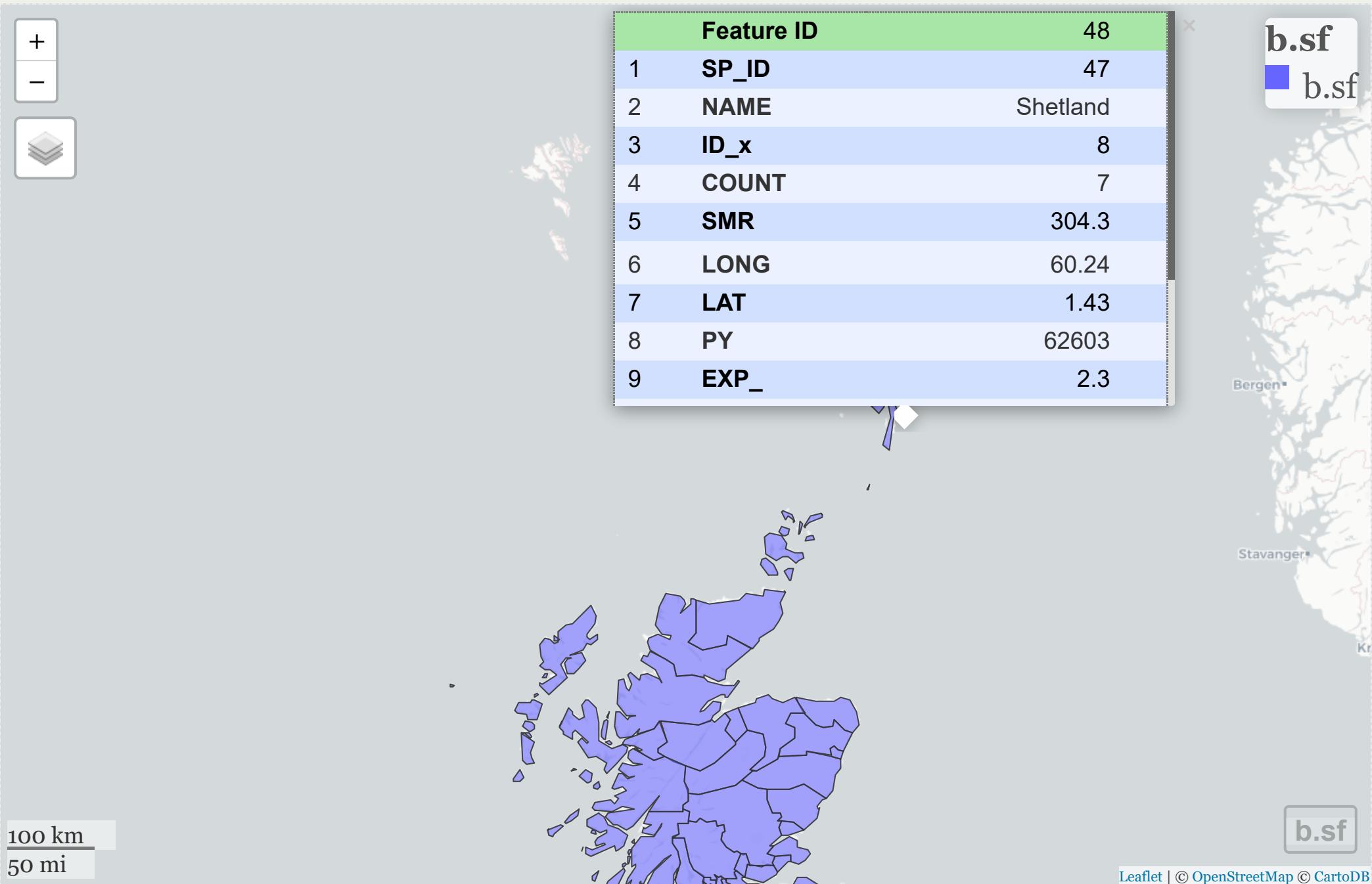
... Поэтому используются возможности других библиотек.

```
require(ggplot2)  
  
ggplot() + geom_sf(data=b.sf, aes(fill=AFF)) + coord_sf(crs=sf::st_crs(3857))
```



Интерактивная визуализация

```
mapview::mapview(b.sf) ## Не отобразится в Jupyter R Notebook.
```



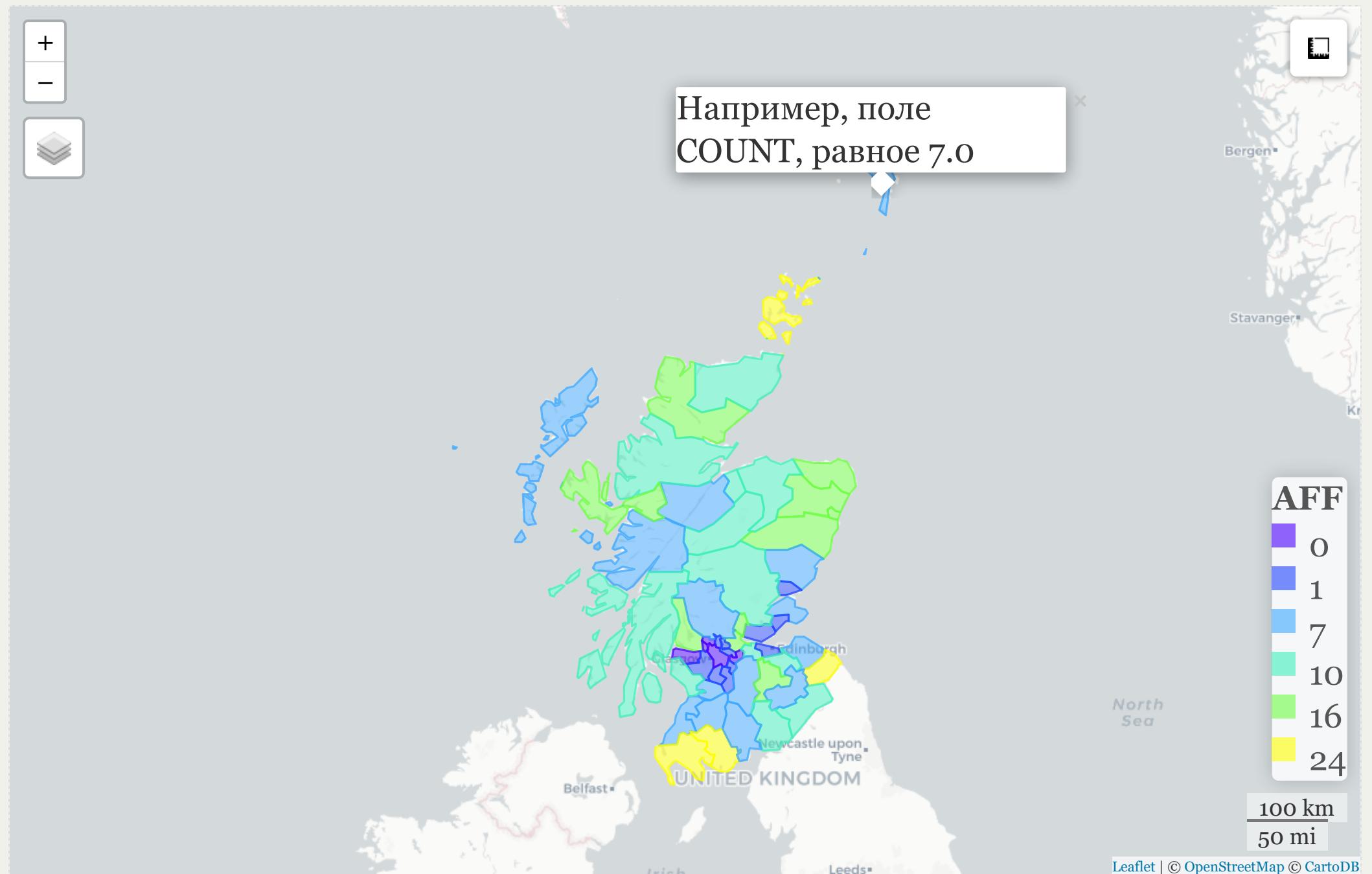
Интерактивная визуализация

```
require(leaflet)

b <- sf:::st_transform(b.sf, 4326)
b$category <- factor(b$AFF, ordered=TRUE)
fpal <- colorFactor(topo.colors(5), b$category)
provList <- c("CartoDB.Positron", "CartoDB.DarkMatter", "Esri.OceanBasemap")
m <- leaflet()
for (p in c(provList)) m <- addProviderTiles(m, providers[[p]], group=p)
m <- m %>%
  addPolygons(data=b, fillColor=~fpal(category), fillOpacity=0.5
              , weight=1.6, color=~fpal(category), opacity=0.75
              , label=~paste0("AFF: ", AFF, " (", NAME, ")"))
  , popup=~sprintf("Например, поле COUNT, равное %.1f", COUNT)
) %>%
  addMeasure("topright", primaryLengthUnit="meters"
             , primaryAreaUnit="sqmeters") %>%
  addScaleBar("bottomright"
              # , options = scaleBarOptions(imperial=FALSE, maxWidth=400)
              ) %>%
  addLayersControl(position="topleft"
                  , baseGroups=c(provList)
                  , options=layersControlOptions(collapsed=TRUE)
                  ) %>%
  addLegend("bottomright", pal=fpal, values=b$category, opacity=0.6
            , title="AFF")
```

Интерактивная визуализация

m ## Не отобразится в Jupyter R Notebook.

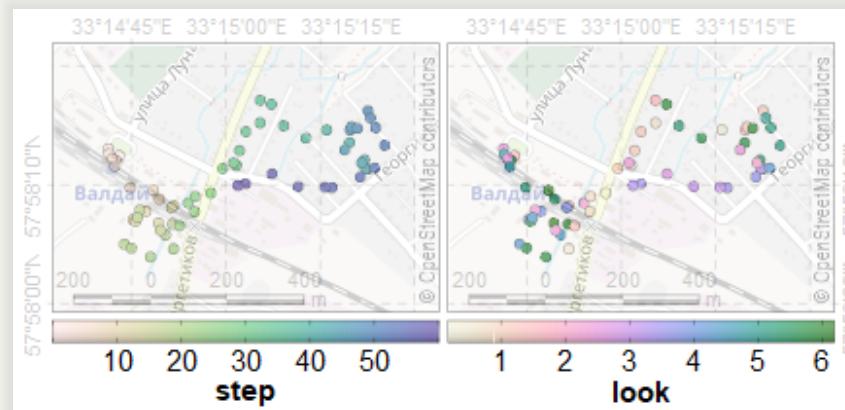


Экспорт пространственных данных

```
pt <- loc  
sp::coordinates(pt) <- ~x+y  
sp::proj4string(pt) <- sp::proj4string(pt0)  
pt <- sp::spTransform(pt, "+init=epsg:4326")  
fileout1 <- "afterTrain.geojson"  
rgdal::writeOGR(pt, fileout1, gsub("\\\\..+", "", basename(fileout1)), driver="GeoJSON",  
    overwrite_layer=TRUE, morphToESRI=FALSE)
```

Проверим, появился ли файл:

```
dir(pattern=paste0(gsub("\\\\..+", "", basename(fileout1)), ".*")))  
[1] "afterTrain.geojson"  
  
try(ursa::glance(fileout1, style="mapnik", las=1, size=200, dpi=99))
```



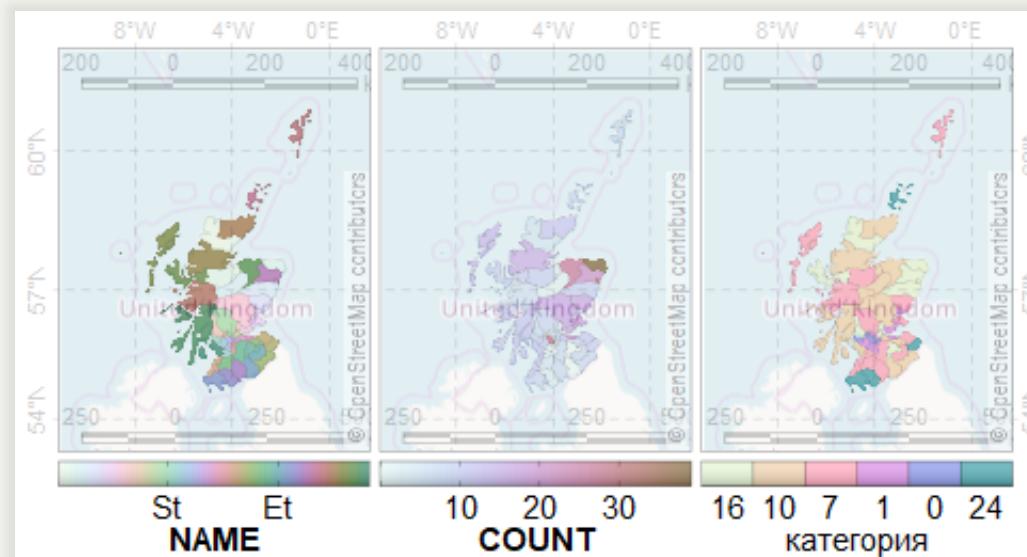
Экспорт пространственных данных

```
b.sf <- b.sf[,c("NAME", "COUNT")]
b.sf$category <- b$category
b.sf <- sf::st_transform(b.sf, 3857)
fileout2 <- "scotland.sqlite"
sf::st_write(b.sf, dsn=fileout2, layer=gsub("\\\\..+", "", basename(fileout2))
             , driver="SQLite", layer_options=c("LAUNDER=NO"), quiet=TRUE
             , delete_layer=file.exists(fileout2), delete_dsn=file.exists(fileout2))

dir(pattern=paste0(gsub("\\\\..+", "", basename(fileout2), ".*")))
[1] "scotland.sqlite"

try(ursa::glance(fileout2, style="mapnik", las=1, dpi=90, size=200))
```

Note: unable to make bold caption for "категория"



Экспорт пространственных данных

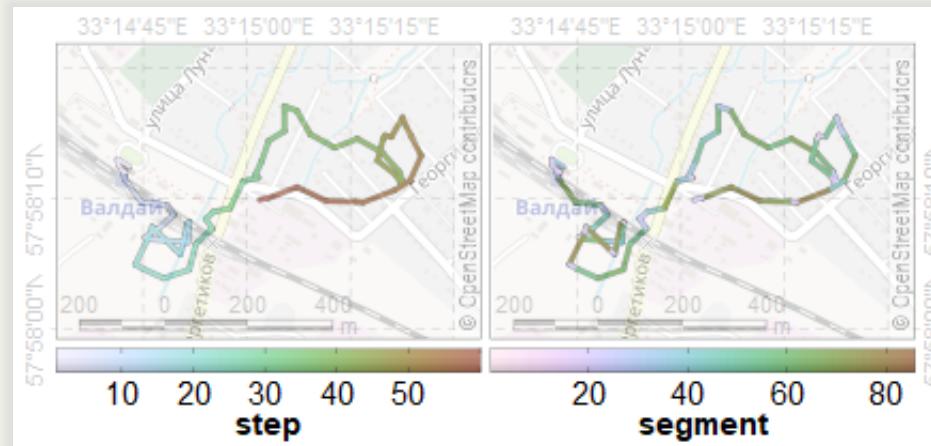
```
fileout3 <- "track.shp"
sf::st_write(tr, dsn=fileout3, layer=gsub("\\\\..+", "", basename(fileout2))
             , driver="ESRI Shapefile", quiet=TRUE

             , delete_layer=file.exists(fileout3), delete_dsn=file.exists(fileout3))

dir(pattern=paste0(gsub("\\\\..+", "", basename(fileout3)), ".*"))

[1] "track.dbf" "track.prj" "track.shp" "track.shx"

try(ursa::glance(fileout3, style="mapnik", las=1, dpi=90, size=200))
```



Рисование

Этот раздел предлагаются пройти самостоятельно

```
track <- sf:::st_linestring(cbind(loc$x, loc$y))
track <- sf:::st_sf(data.frame(desc="walk"
                               ,sf:::st_sfc(track, crs=sp::proj4string(pt0)) ))
paint <- mapview::viewExtent(track, alpha=0.01) %>% mapedit::editMap("track")
result <- NULL
if (!is.null(paint$finished)) {
  result <- paint$finished
  mapview::mapview(result)
  ursa::session_grid(NULL)
  ursa::glance(result, style="mapnik")
}
```

Воспроизводимые вычисления и публикация результата

Ниже приведены фрагменты кода, которые не были выполнены при составлении программы занятия. Их предлагается выполнить самостоятельно. Также необходимо установленный Pandoc.

Содержимое занятия сгенерировано из файла `main.R`. Загрузим его, переименовав:

```
sfile <- "main.R"
rfile <- "lesson.R"
{
  if (file.exists(sfile))
    file.copy(sfile,rfile,overwrite=TRUE,copy.date=TRUE)
  else
    download.file(file.path("https://nplatonov.github.io/SCGIS2019", sfile)
                  , rfile)
}
```

Воспроизводимые вычисления и публикация результата

На основе этого файла создадим markdown-документ:

```
mdfile <- knitr::spin(rfile, knit=FALSE)
mdfile
```

Затем из markdown-документа сформируем html-документ:

```
if (rmarkdown::pandoc_available()) {
  htmlfile <- rmarkdown::render(mdfile, output_format="html_document"
                               , output_options=list(toc=TRUE))
  print(basename(htmlfile))
}
```

Полученный html-документ можно открыть в браузере.

```
if ((rmarkdown::pandoc_available() ) && (file.exists(htmlfile) ))
  browseURL(basename(htmlfile))
```

И можно скопировать на флешку на память:))

Успехов!

Дополнительная информация

Как узнать об R поглубже

- R-Bloggers – современные тенденции R
- R's Spatial and SpatioTemporal Task Views
- R-Spatial – веб-сайт и блог для интересующихся использованием R для анализа пространственных и пространственно-временных данных
- Stackoverflow с тегом #R.
- Stackexchange о ГИС, в т.ч. с использованием R.
- Package's vignettes – обобщенное знакомство с библиотекой. Обычно содержат воспроизводимый код.

Ведущий

- Платонов Никита Геннадьевич  
- ИПЭЭ РАН – 85 лет в 2019 г. 
- Постоянно действующая экспедиция РАН по изучению животных Красной книги Российской Федерации и других особо важных животных фауны России
- Программа изучения белого медведя в Российской Арктике

Данные после занятия остались на диске. Если не нужны, выполнить следующее:

```
file.remove(dir(pattern=paste0(gsub("\\" . "+" , "", basename(fileout1)), "."))
file.remove(dir(pattern=paste0(gsub("\\" . "+" , "", basename(fileout2)), "."))
file.remove(dir(pattern=paste0(gsub("\\" . "+" , "", basename(fileout3)), ".")
```