

Operations With Raster Data I

HES 505 Fall 2022: Session 11

Matt Williamson



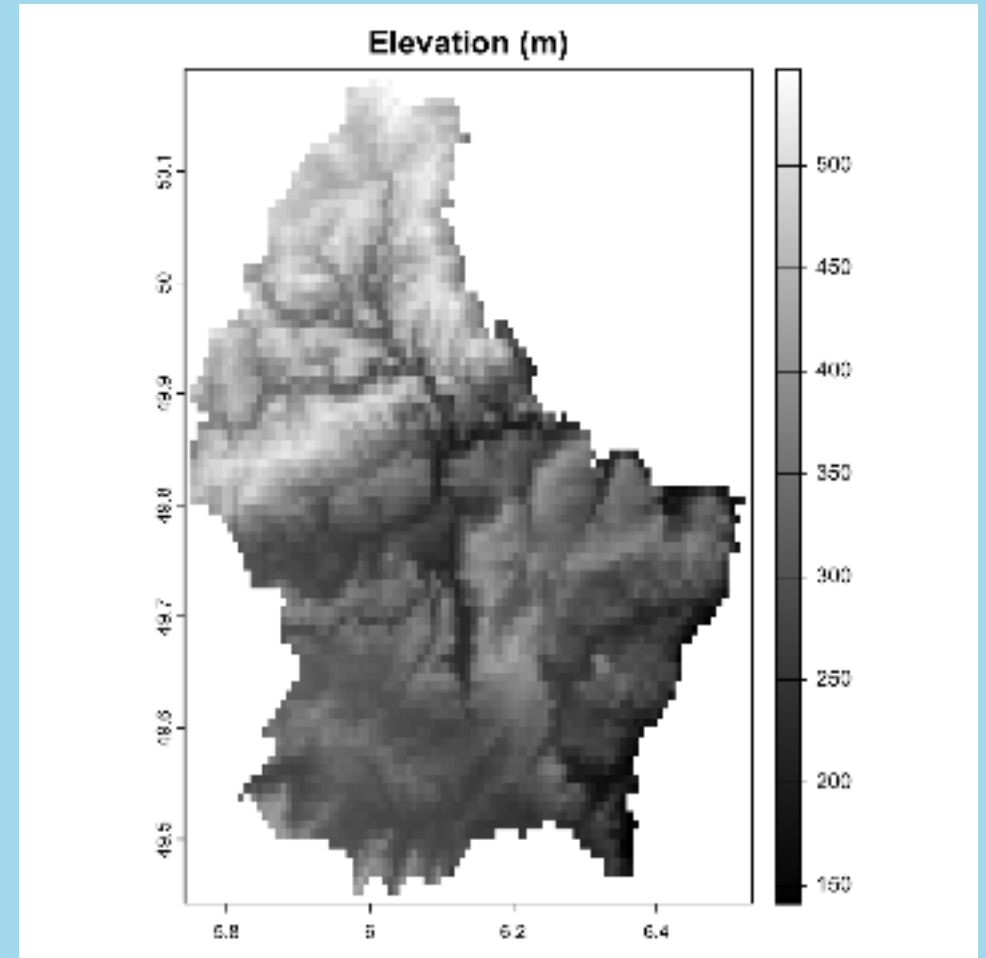
Today's Plan

Objectives

- By the end of today, you should be able to:
 - Evaluate logical conditions with raster data
 - Calculate different measures of raster data
 - Align rasters for spatial processing

Revisiting the Raster Data Model

- Raster data represent spatially continuous phenomena (NA is possible)
- Depict the alignment of data on a regular lattice (often a square)
 - Operations mimic those for `matrix` objects in R
- Geometry is implicit; the spatial extent and number of rows and columns define the cell size



Predicates and measures in **terra**

Extending predicates

- **Predicates:** evaluate a logical statement asserting that a property is **TRUE**
- **terra** does not follow the same hierarchy as **sf** so a little trickier

Unary predicates in terra

- Can tell us qualities of a raster dataset
- Many similar operations for **SpatVector** class (note use of **.**)

predicate	asks...
is.lonlat	Does the object have a longitude/latitude CRS?
inMemory	is the object stored in memory?
is.factor	Are there categorical layers?
hasValues	Do the cells have values?

Unary predicates in **terra**

- **global**: tests if the raster covers all longitudes (from -180 to 180 degrees) such that the extreme columns are in fact adjacent
- **perhaps**: If TRUE and the crs is unknown, the method returns TRUE if the coordinates are plausible for longitude/latitude

```
1 r <- rast()  
2 is.lonlat(r)
```

```
[1] TRUE
```

```
1 is.lonlat(r, global=TRUE)
```

```
[1] TRUE
```

```
1 crs(r) <- ""  
2 is.lonlat(r)
```

```
[1] NA
```

```
1 is.lonlat(r, perhaps=TRUE, warn=FALSE)
```

```
[1] TRUE
```

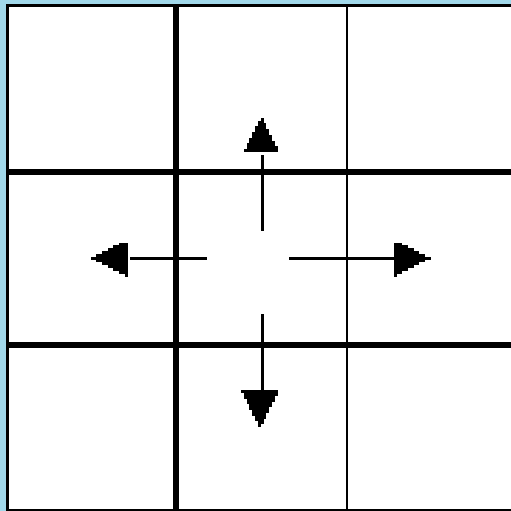
```
1 crs(r) <- "+proj=lcc +lat_1=48 +lat_2=33 +lon_0=-1  
2 is.lonlat(r)
```

```
[1] FALSE
```

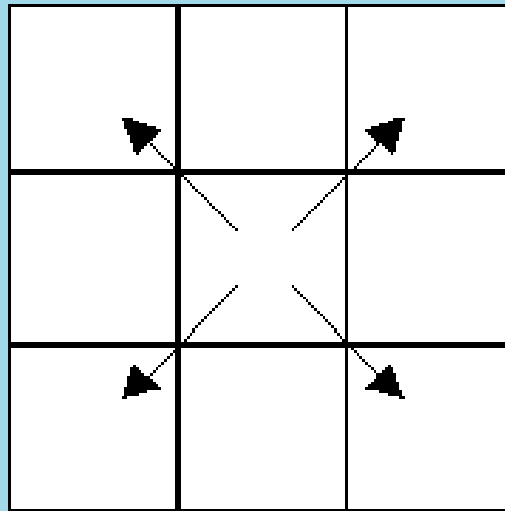

Binary predicates in **terra**

- Take exactly 2 inputs, return 1 matrix of cell locs where value is **TRUE**
- **adjacent**: identifies cells adjacent to a set of raster cells

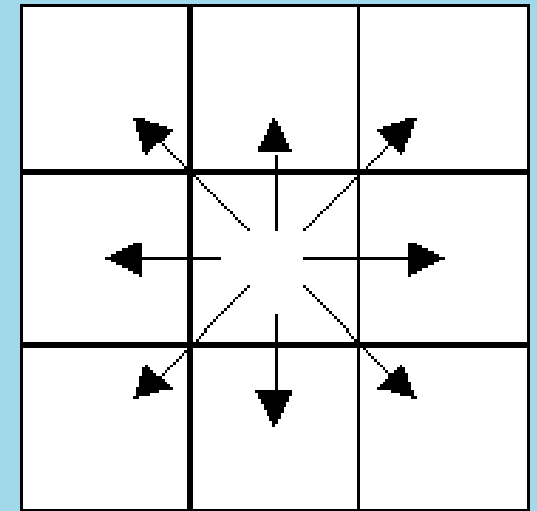
Rooks Case



Bishops Case



Queen's (Kings) Case



Unary measures in terra

- Slightly more flexible than sf
- One result for each layer in a stack

measure	returns
cellSize	area of individual cells
expanse	summed area of all cells
values	returns all cell values
ncol	number of columns
nrow	number of rows
ncell	number of cells
res	resolution
ext	minimum and maximum of x and y coords
origin	the origin of a SpatRaster
crs	the coordinate reference system
cats	categories of a categorical raster

Binary measures in **terra**

- Returns a matrix or **SpatRaster** describing the measure

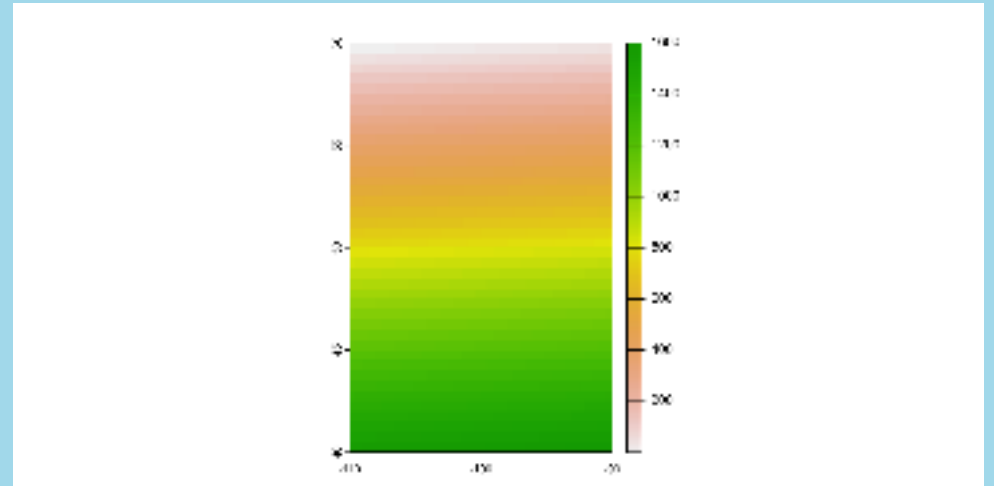
measure	returns
distance	shortest distance to non-NA or vector object
gridDistance	shortest distance through adjacent grid cells
costDistance	Shortest distance considering cell-varying friction
direction	azimuth to cells that are not NA

Aligning rasters

Projecting raster data

- Transformation from lat/long to planar CRS involves some loss of precision
- New cell values estimated using overlap with original cells
- Interpolation for continuous data, nearest neighbor for categorical data
- Equal-area projections are preferred; especially for large areas

```
1 r <- rast(xmin=-110, xmax=-90, ymin=40, ymax=60, n  
2 values(r) <- 1:ncell(r)  
3 plot(r)
```

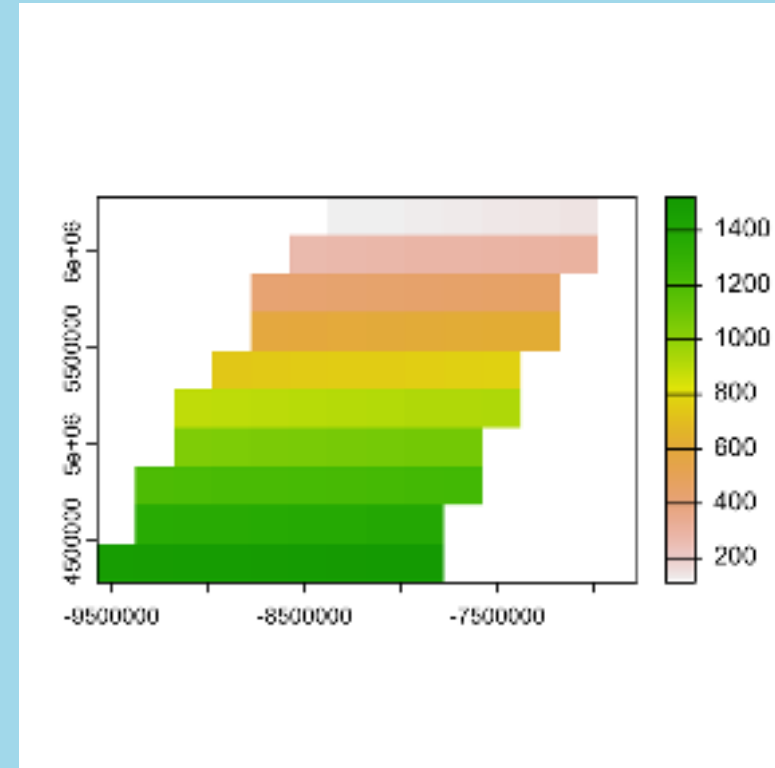
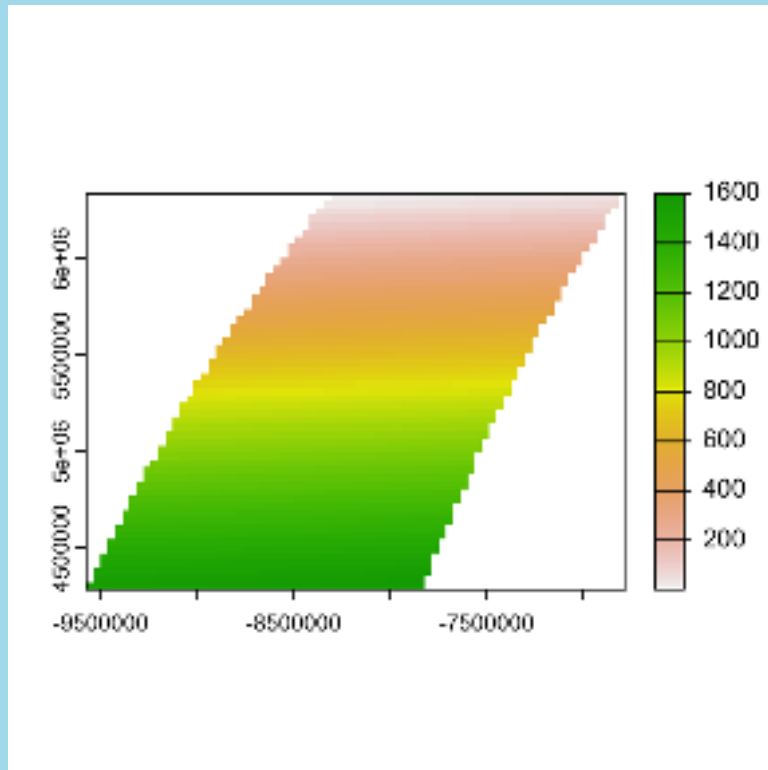


Projecting raster data

Projecting raster data

- simple method; alignment not guaranteed
- providing a template to ensure alignment

```
1 newcrs <- "+proj=robin +datum=WGS84"
2 pr1 <- terra::project(r, newcrs)
3 plot(pr1)
```

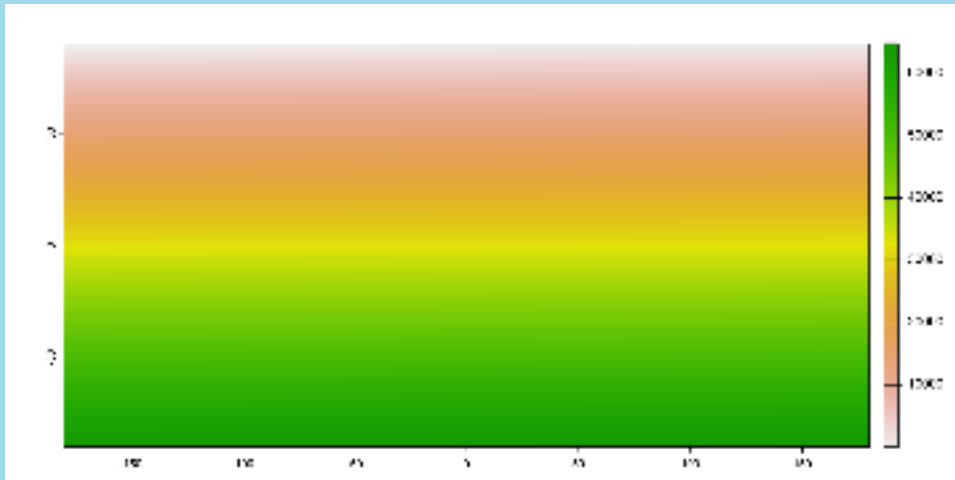


Changing resolutions

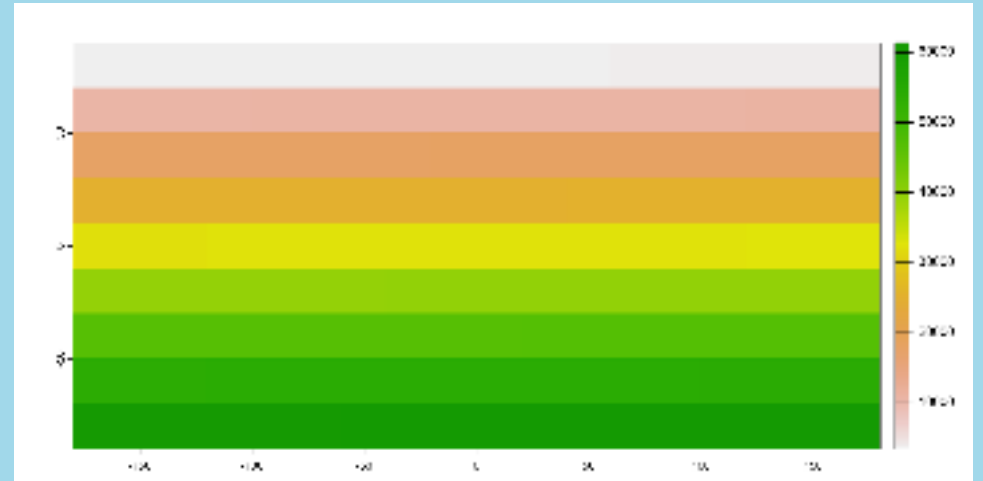
- **aggregate**, **disaggregate**, **resample** allow changes in cell size
- **aggregate** requires a function (e.g., **mean()** or **min()**) to determine what to do with the grouped values
- **resample** allows changes in cell size **and** shifting of cell centers (slower)

Changing resolutions: aggregate

```
1 r <- rast()  
2 values(r) <- 1:ncell(r)  
3 plot(r)
```



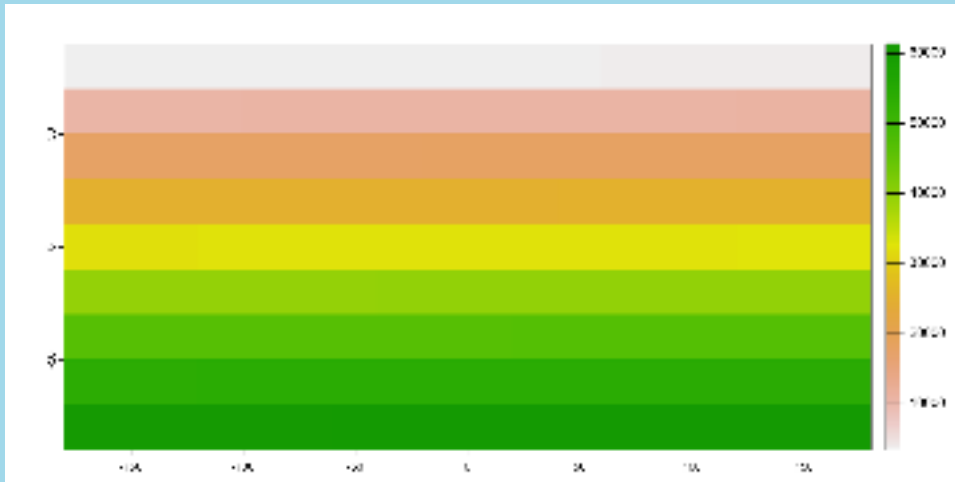
```
1 ra <- aggregate(r, 20)  
2 plot(ra)
```



]

Changing resolutions: disagg

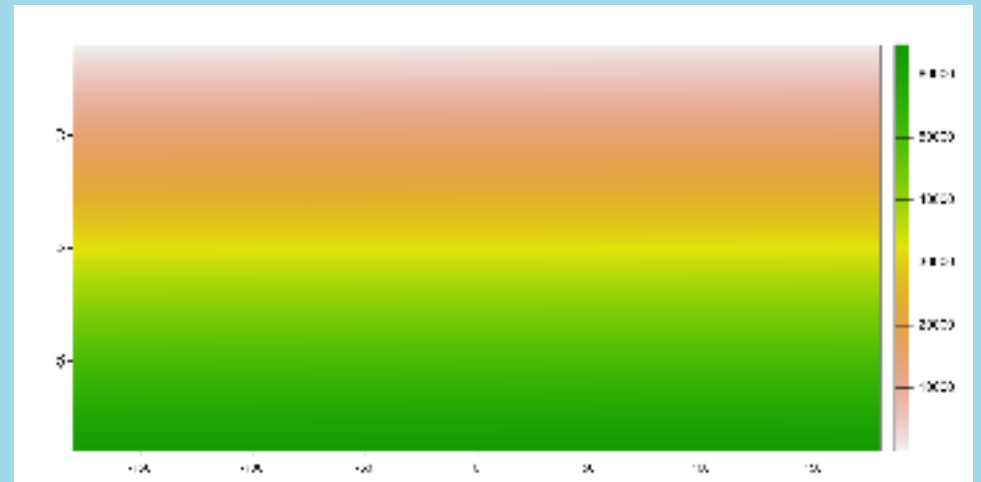
```
1 ra <- aggregate(r, 20)  
2 plot(ra)
```



```
1 rd <- disagg(r, 20)
```

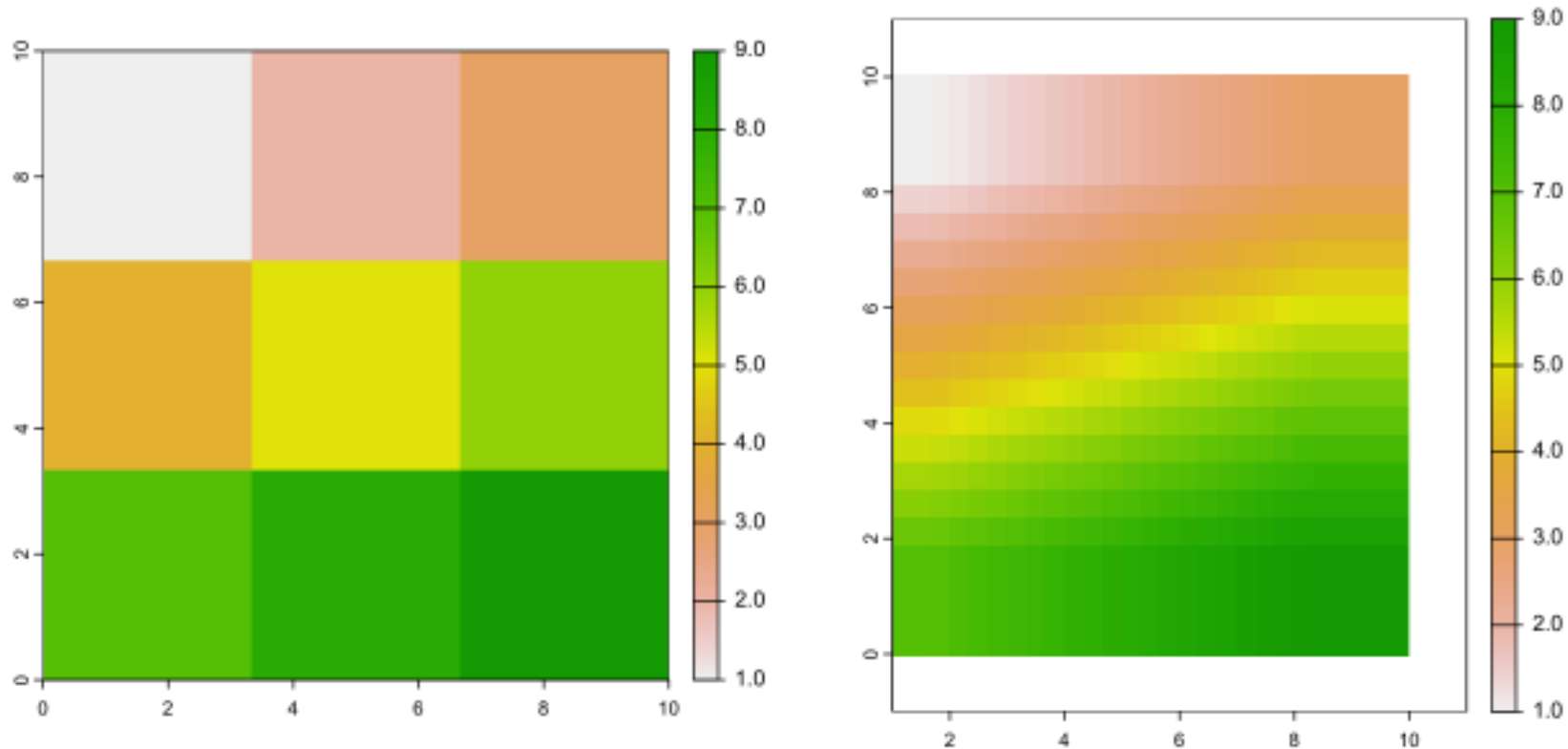
```
|-----|-----|-----|-----|  
--|  
====
```

```
1 plot(rd)
```



Changing Resolutions: resample

```
1 r <- rast(nrow=3, ncol=3, xmin=0, xmax=10, ymin=0, ymax=10)
2 values(r) <- 1:ncell(r)
3 s <- rast(nrow=25, ncol=30, xmin=1, xmax=11, ymin=-1, ymax=11)
4 x <- resample(r, s, method="bilinear")
```



On Weds

- Transformations of data and coverage
- Raster math
- Cell-based functions

