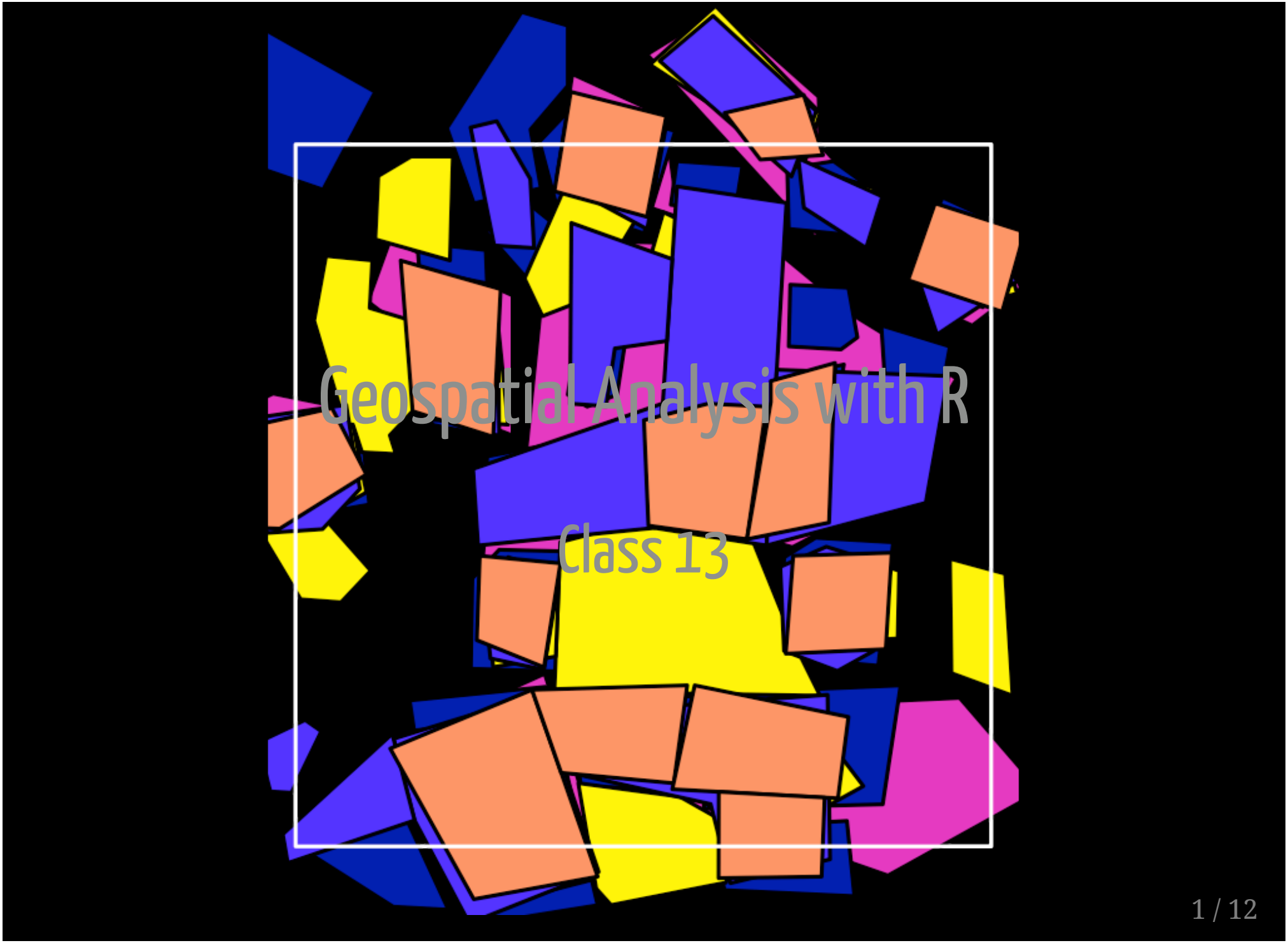
Geospatial Analysis with R
Class 13

Today

- Team-based practicals:
  - Data preparation
  - Data analysis

Tips

- Shortcuts: CTRL + Alt + i; CMD/ctrl + shift + enter; CMD/ctrl + alt + n
- Chunk settings:

````
```{r, eval=FALSE, message=FALSE, warning=FALSE, error=TRUE, echo=FALSE}
# Your code in here
```
````

- Pay attention to style guide and syntax!!!
    - It will help minimize coding errors

Syntax dos and don'ts

```r
a <- 1:10  # yes
a<-1:10  # no (you can't make this if you use crtl/CMD + -)
a <-1:10  # no (you can't make this if you use crtl/CMD + -)
a<- 1:10  # no (you can't make this if you use crtl/CMD + -)

lapply(1:10, function(x) {  # yes
  print(x)
})
lapply (1:10,function(x){  # no
print(x)
})

dat %>% filter(a > 10) %>% mutate(a = a * 10)  # yes
dat%>%filter (a>10)%>%mutate (a=a*10)  # no

mean(x = c(NA, 1:10), na.rm = TRUE)  # yes
mean (x=c(NA,1:10),na.rm=T)  # no

dat <- data.frame(v1 = 1:10, v2 = 1:10)  # yes
dat<-data.frame(v1=1:10,v2=1:10)  # certainly no
dat$v1  # yes
dat $v1  # no
dat$ v1 # no
dat $ v1 # no
dat       $      v1  # hell no
```

Code elements

```
# 1
p <- "~/Desktop/dummy_dataset.csv"
# 2
readr::read_csv()
# 3
spread()
# 4
mutate()
# 5
dat2 <- dat
# 6
select()
# 7
arrange()
# 8
desc()
# 9
group_by()
# 10
ifelse()
# 11
filter()
```

Code elements continued

```r
# 12
lapply(1:nrow(dat), function(x) {
  # code here
})
# 13
nrow()
# 14
slice()
# 15
for(i in 1:nrow(dat) {
  # code here
}
# 16
summarize()
# 17
summarize_all()
# 18
list(); funs()
# 19 - a full pipeline example
library(dplyr)
library(tidyr)
dat <- readr::read_csv("~/Desktop/dummy_dataset.csv") %>%
  spread(element, value) %>%
  mutate(wt_price = Weight / Price) %>%
  select(group, Price, Weight, wt_price) %>%
  group_by(group) %>%
  # summarise_all(funs(mean, sd))  # this works but is deprecated
  summarise_all(list(mean, sd))
```

Practical 1 - data preparation

- Read `dummy_dataset.csv` into `dat` (elements #1 and #2)
- Make a new tidy `tibble` `dat2`, which has "Price" and "Weight" in their own columns (#3, #5)
- Update `dat2` so that it has a new column *wt_price* from `weight / price` (#4, #5)
- Update `dat2` so that it is rearranged by *year* from oldest to most recent and by *Price* from cheapest to most expensive (#5, #7)
- Update `dat2` so that is has a new column *value*, which lists *wt_price* values <1 as "cheap" and values >1 as "pricy" (#4, #5, #10)
- Create `dat3`, redoing all the steps above, but within a single pipeline (#19)

Practical 1 answers

Buried in the Rmarkdown

```
dat <- readr::read_csv("~/Desktop/dummy_dataset.csv")
dat2 <- dat %>% spread(key = element, value = value)
dat2 <- dat2 %>% mutate(wt_price = Weight / Price)
dat2 <- dat2 %>% arrange(year, Price)
dat2 <- dat2 %>% mutate(value = ifelse(wt_price < 1, "cheap", "pricy"))
dat3 <- readr::read_csv("~/Desktop/dummy_dataset.csv") %>%
  spread(key = element, value = value) %>%
  mutate(wt_price = Weight / Price) %>%
  arrange(year, Price) %>%
  mutate(value = ifelse(wt_price < 1, "cheap", "pricy"))
```

Practical 2 - looping

- Create a new 2-element list `datl` from `dat3`. Use the vector `c("cheap", "pricy")` as your iterator in an `lapply`, selecting (`filter`) on the new *value* variable within the `lapply` to subset the dataset by value (#5, #11, #12)
- Use a `for` loop to iterate over the rows of `dat3`, using `slice` to pull out each row (i), `select`ing just *Price* and *Weight* from the row. `print` it. (#6, #13-15)

Practical 2 answers

In the Rmarkdown

```
datl <- lapply(c("cheap", "pricy"), function(x) {
  dat2 %>% filter(value == x)
})
names(datl) <- c("cheap", "pricy")
for(i in 1:nrow(dat3)) print(dat3 %>% slice(i) %>% select(Price, Weight))
```

Practical 3 - data analysis

- From `dat3` calculate the `mean` *wt_price* by *group* (#9, #16)
- From `dat3` calculate the `mean` and `sd` of *wt_price* by *group* (#9, #16)
- From `dat3` calculate the `mean`, `sd` of *Weight* and *Price* by *group* and *value* (#9, #16)
- From `dat3` select *Weight*, *Price*, *wt_price* and calculate the `mean`, `sd` of all three variables at once (#9, #17-19)
- Do the same as above, but do summarize by value (#6, #9, #17-19)
- From `dat3`, extract just the cheap *value*s, and calculate the *group*-wise `mean` of *Weight*, *Price*, *wt_price* (#6, #11, #17)
- **Challenge**: Use a `dplyr` pipeline to fit a regression (`lm`) between *Weight* and *Price*. Get the `summary()` of the `lm` from the tail-end of the pipeline
- **Even more challenge**. Use a pipeline with `do`, `lm`, and `broom::tidy` to create and output the coefficients of regressions on "cheap" and "pricy" values (see Chunk 39 in Unit 1 - Module 4)

Practical 3 answers

Buried in the Rmarkdown