CEBRA

PL

11/07/2017

knitr::opts\_chunk$set(echo = TRUE, warning=FALSE, message=FALSE)  
library(sp)  
library(leaflet)  
library(rgeos)

## rgeos version: 0.3-23, (SVN revision 546)  
## GEOS runtime version: 3.4.2-CAPI-1.8.2 r3921   
## Linking to sp version: 1.2-4   
## Polygon checking: TRUE

library(plyr)  
library(readxl)  
library(rgdal)

## rgdal: version: 1.2-8, (SVN revision 663)  
## Geospatial Data Abstraction Library extensions to R successfully loaded  
## Loaded GDAL runtime: GDAL 2.1.2, released 2016/10/24  
## Path to GDAL shared files: /Users/grant/Library/R/3.3/library/rgdal/gdal  
## Loaded PROJ.4 runtime: Rel. 4.9.1, 04 March 2015, [PJ\_VERSION: 491]  
## Path to PROJ.4 shared files: /Users/grant/Library/R/3.3/library/rgdal/proj  
## Linking to sp version: 1.2-5

library(PBSmapping)

##   
## -----------------------------------------------------------  
## PBS Mapping 2.70.4 -- Copyright (C) 2003-2017 Fisheries and Oceans Canada  
##   
## PBS Mapping comes with ABSOLUTELY NO WARRANTY;  
## for details see the file COPYING.  
## This is free software, and you are welcome to redistribute  
## it under certain conditions, as outlined in the above file.  
##   
## A complete user guide 'PBSmapping-UG.pdf' is located at   
## /Users/grant/Library/R/3.3/library/PBSmapping/doc/PBSmapping-UG.pdf  
##   
## Packaged on 2017-06-28  
## Pacific Biological Station, Nanaimo  
##   
## All available PBS packages can be found at  
## https://github.com/pbs-software  
##   
## To see demos, type '.PBSfigs()'.  
## -----------------------------------------------------------

library(png)  
library(grid)  
library(mapview)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:rgeos':  
##   
## intersect, setdiff, union

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(utils)  
library(tidyr)  
library(magrittr)

##   
## Attaching package: 'magrittr'

## The following object is masked from 'package:tidyr':  
##   
## extract

library(RColorBrewer)  
library(tmap)  
library(ellipse)  
require(lattice)

## Loading required package: lattice

library(ggplot2)

## Summary

This documents the data, the code and the process behind the CEBRA app.

## Data

The data on the boundaries of regions in Australia comes from [Australian Bureau of Statistics](http://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/1270.0.55.001July%202011?OpenDocument). Dowload "Statistical Area Level 2 (SA2) ASGS Ed 2011 Digital Boundaires in ESRI Shapefile Format" into your home directory. Find more about statistical regions of Australia [here](http://stat.abs.gov.au/itt/r.jsp?ABSMaps).

Use a specific type of projection in order to calculate the area consistently [proj4js](http://spatialreference.org/ref/sr-org/australia-albers-equal-area-conic-134/proj4js/) in square meters. I validated that this projection method works with "gArea" using East Pilbara Shire which has an area of about 380,000 km².

#Read the file with regional boundaries into a var "Aust"  
 #Get the data   
 file<-"SA2\_2011\_AUST.shp"  
 Aust<-readOGR(file, GDAL1\_integer64\_policy = TRUE)

## OGR data source with driver: ESRI Shapefile   
## Source: "SA2\_2011\_AUST.shp", layer: "SA2\_2011\_AUST"  
## with 2214 features  
## It has 12 fields

#Transform the coord system to a specific type which will be useful later  
 proj\_Aust<-  
 "+proj=aea +lat\_1=-18 +lat\_2=-36 +lat\_0=0 +lon\_0=134 +x\_0=0 +y\_0=0 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no\_defs"  
  
 #Generic projection  
 proj<- "+proj=longlat +datum=WGS84"  
  
 Aust <- spTransform(Aust, CRS(proj\_Aust))  
 #Check the plot  
 #plot(Aust)  
  
 #Add a data column with area calculated by gArea in m^2, divide by 10000 to get ha  
 Aust@data<-mutate(Aust@data, Area\_ha=gArea(Aust,byid=TRUE)/10000)  
  
 #Look at the top 4 rows in order of area   
 head(Aust@data[order(-Aust@data$Area\_ha),], 4)

## SA2\_MAIN11 SA2\_5DIG11 SA2\_NAME11 SA3\_CODE11  
## 1638 406021141 41141 Outback 40602  
## 1870 508031203 51203 Leinster - Leonora 50803  
## 1882 508051215 51215 Meekatharra 50805  
## 1886 508061219 51219 East Pilbara 50806  
## SA3\_NAME11 SA4\_CODE11 SA4\_NAME11  
## 1638 Outback - North and East 406 South Australia - Outback  
## 1870 Goldfields 508 Western Australia - Outback  
## 1882 Mid West 508 Western Australia - Outback  
## 1886 Pilbara 508 Western Australia - Outback  
## GCC\_CODE11 GCC\_NAME11 STE\_CODE11 STE\_NAME11 ALBERS\_SQM  
## 1638 4RSAU Rest of SA 4 South Australia 519519951490  
## 1870 5RWAU Rest of WA 5 Western Australia 496740603473  
## 1882 5RWAU Rest of WA 5 Western Australia 414505958216  
## 1886 5RWAU Rest of WA 5 Western Australia 389540916479  
## Area\_ha  
## 1638 51951995  
## 1870 49674060  
## 1882 41450596  
## 1886 38954092

#Look at the names of the 10 largest by area regions   
 unique(Aust@data[order(-Aust@data$Area\_ha),][1:10,][,"SA2\_NAME11"])

## [1] Outback Leinster - Leonora   
## [3] Meekatharra East Pilbara   
## [5] Barkly Far Central West   
## [7] Kambalda - Coolgardie - Norseman Tanami   
## [9] Far South West Petermann - Simpson   
## 2214 Levels: Abbotsford Aberfoyle Park ACT - East ACT - South West ... Zillmere

### Populating the dataframe with real data

The first step is to populate regions with statistics on population. Dowload "Population Estimates by Age and Sex, Summary Statistics (ASGS 2011), 2010 and 2015" excel file from [Australian Bureau of Statistics](http://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/3235.02015?OpenDocument).

Extract populations numbers by region from this data file and merge it with "Aust" dataframe. Not every SA2 region has a population entry, but in total about 22 mil are accounted for, and according to spot check (East Pilbara) and the map seems to have been recorded in the correct regions in the dataframe.

#Name of the file as downloaded   
 file<-"32350ds0009\_sa2\_summary\_statistics\_2010\_2015.xls"  
 #Read the file  
 pop\_regions <- read\_excel(file, sheet="Table 1", skip=15)[1:3544,c(6,9)] %>% setNames(  
 c("SA2\_NAME11","Population"))  
 #Get rid of rows without names  
 pop\_regions<-subset(pop\_regions, !is.na(pop\_regions$SA2\_NAME11))  
  
 #Check classes of columns  
 class(pop\_regions$Population)

## [1] "numeric"

class(pop\_regions$SA2\_NAME11)

## [1] "character"

#Check that all regions are distinct  
 length(pop\_regions$SA2\_NAME11)==length(unique(pop\_regions$SA2\_NAME11))

## [1] TRUE

#Merge population data with regions  
 Aust<-merge(Aust, pop\_regions, by="SA2\_NAME11")  
  
 #Check how many people are accounted for  
 sum(Aust$Population, na.rm = TRUE)

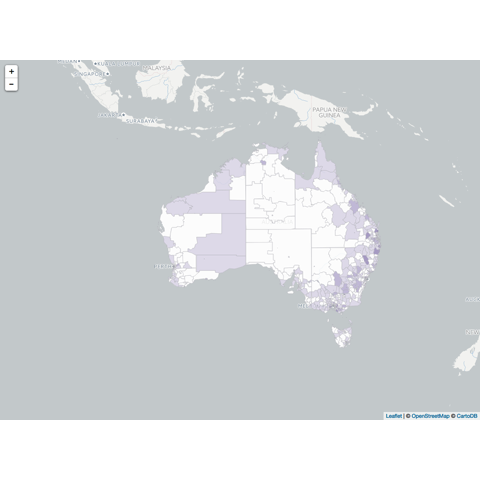
## [1] 22015719

Make a map image with the population data.

#Set eval to false because it takes a while  
 #Plot  
 pal\_pop<- colorBin(c("white","darkslateblue"), domain= Aust$Population)  
  
 map\_pop<-leaflet(data = spTransform(Aust,CRS(proj))) %>% addProviderTiles("CartoDB.Positron") %>%  
 addPolygons(fillColor = ~pal\_pop(Population), # refers to pal defined above  
 fillOpacity = 0.8,   
 color = "#BDBDC3", # colour between polygons  
 weight = 1,  
 highlight = highlightOptions(  
 weight = 5,  
 color = "pink",  
 dashArray = "",  
 fillOpacity = 0.7,  
 bringToFront = TRUE))  
  
 mapshot(map\_pop, file = "Map\_Population.png")

Show where the population is recorded in the "Aust" dataframe:

img <- readPNG("Map\_Population.png")  
 grid.raster(img)



### Subsetting the regional data

There are various ways to subset the regional data, for example by name, by area and by geographical location.The code below finds the names of the administrative regions that lie within four boxes defined by latitude and longitude.

We first define a matrix of coordinates for the box, then turn it into an R object of the "SpatialPolygonsDataFrame" class, with the the same projection method (CRS) as "Aust", so that we can use the function "over" to create an index indicating the overlap between the box and administrative regions. Based on that index, we create a list of names for the regions near the coasts.

The slight complication is that we define the boxes in one global projection system, and then translate them into the Australia specific one, using "spTransform".

Why are we doing this? In case we want to simulate random but plausible distribution of pests (i.e. near coasts).

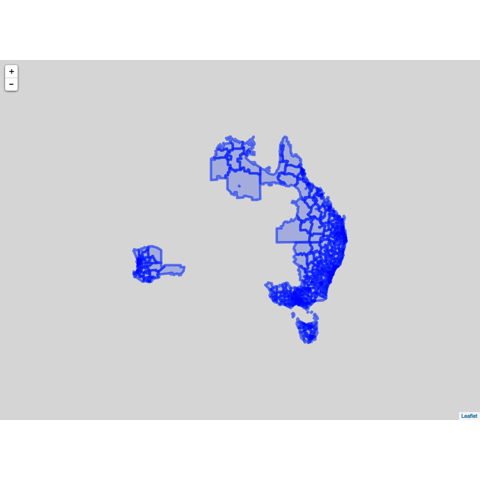
#Find the regions in the box  
 boxSW<-cbind(c(114,114,120,120,114),c(-31,-36,-36,-31,-31))  
 p = Polygon(boxSW)  
 ps = Polygons(list(p), 1)  
 sps = SpatialPolygons(list(ps))  
 proj4string(sps) = CRS(proj)  
 sps = spTransform(sps, CRS(proj\_Aust))  
 data = data.frame(f=99.9)  
 spdf = SpatialPolygonsDataFrame(sps,data)  
 indexSW<-over(Aust, spdf)  
 regionsSW<-unique(Aust@data[!is.na(indexSW),]$SA2\_NAME11)  
  
 boxSE<-cbind(c(138,138,150,150,138),c(-36,-48,-48,-36,-36))  
 p = Polygon(boxSE)  
 ps = Polygons(list(p), 1)  
 sps = SpatialPolygons(list(ps))  
 proj4string(sps) = CRS(proj)  
 sps = spTransform(sps, CRS(proj\_Aust))  
 data = data.frame(f=99.9)  
 spdf = SpatialPolygonsDataFrame(sps,data)  
 indexSE<-over(Aust, spdf)  
 regionsSE<-unique(Aust@data[!is.na(indexSE),]$SA2\_NAME11)  
  
 boxEast<-cbind(c(146,146,155,155,146),c(-36,-18,-18,-36,-36))  
 p = Polygon(boxEast)  
 ps = Polygons(list(p), 1)  
 sps = SpatialPolygons(list(ps))  
 proj4string(sps) = CRS(proj)  
 sps = spTransform(sps, CRS(proj\_Aust))  
 data = data.frame(f=99.9)  
 spdf = SpatialPolygonsDataFrame(sps,data)  
 indexEast<-over(Aust, spdf)  
 regionsEast<-unique(Aust@data[!is.na(indexEast),]$SA2\_NAME11)  
  
 boxNorth<-cbind(c(132,132,155,155,132),c(-12,-18,-18,-12,-12))  
 p = Polygon(boxNorth)  
 ps = Polygons(list(p), 1)  
 sps = SpatialPolygons(list(ps))  
 proj4string(sps) = CRS(proj)  
 sps = spTransform(sps, CRS(proj\_Aust))  
 data = data.frame(f=99.9)  
 spdf = SpatialPolygonsDataFrame(sps,data)  
 indexNorth<-over(Aust, spdf)  
 regionsNorth<-unique(Aust@data[!is.na(indexNorth),]$SA2\_NAME11)

Now that we have names of the regions from which we want to simulate pest invasion and industry data, we can subset the "Aust" object to focus on those regions.

Aust\_Coast <-subset(Aust,  
 (Aust@data[,"SA2\_NAME11"] %in% regionsSW)|(  
 Aust@data[,"SA2\_NAME11"] %in% regionsSE)|  
 (Aust@data[,"SA2\_NAME11"] %in% regionsEast)|  
 (Aust@data[,"SA2\_NAME11"] %in% regionsNorth))  
  
 #To see the map of the selected regions  
 map<-leaflet(data = spTransform(Aust\_Coast,CRS(proj))) %>% addPolygons()  
 mapshot(map, file = "Map\_Coast\_Aust.png")

Show map of selected regions:

img <- readPNG("Map\_Coast\_Aust.png")  
 grid.raster(img)



### Data on industries

Data for various agricultural industries is available on SA2 scale, download csv file "Agricultural commodities, Australia, state, territory and SA2 - (Data by estimated value of agricultural operations class)" from [Australian Bureau of Statistics](http://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/7121.02010-11?OpenDocument).

This data requires quite a bit of cleaning. It is very detailed containing 549 categories of observations on SA2 scale. Apparently, only aggregated data on $ values is available, [Total Values](http://www.abs.gov.au/AUSSTATS/abs@.nsf/allprimarymainfeatures/E66BFDC17D5C631DCA25815500125AA2?opendocument), but there is very detailed data on the areas of cultivation, and yield in weight per ha, etc. For instance, if someone is interested in fennel... the data is there.

We will extract approximate information on distribution of several key commodities: Broadacre crops, Hay, Flowers, Fruit and Nuts and Grapes, Vegetables, and Livestock. Then we merge it with spatial dataframe Aust, and make a visual map of each. The units are ha.

This is not a precise operation and should be viewed as indicative of distribution because there are some unanswered questions about the dataset. For example, several values for the same region show up under the same category - I decided to add them, speculating that these are subregional data, but another explanation is possible. I did not check.

#Read the file  
 file<-"71210do042\_201011.csv"  
 agri\_regions <- read.csv(file, skip=4)  
 #Select only relevant columns  
 agri\_regions <-select(agri\_regions, c(2,6,7))  
 names(agri\_regions)<- c("SA2\_NAME11","Commodity","Value")  
 #Make sure they are in the right format  
 agri\_regions$SA2\_NAME11<-as.character(agri\_regions$SA2\_NAME11)  
 agri\_regions$Value<-as.numeric(agri\_regions$Value)  
 agri\_regions$Commodity<-as.character(agri\_regions$Commodity)  
 #Only keep those regions which match our existing dataset  
 agri\_regions <-filter(agri\_regions, agri\_regions$SA2\_NAME11 %in% unique(Aust@data[,"SA2\_NAME11"]))  
 #Get rid of NA values in areas of production  
 agri\_regions <-filter(agri\_regions, !is.na(agri\_regions$Value))  
 #Get rid of small values  
 agri\_regions <-filter(agri\_regions, agri\_regions$Value > 100)  
 #Get rid of duplicate rows if any  
 agri\_regions<-distinct(agri\_regions)  
  
 #This is how many categories there are  
 length(unique(agri\_regions$Commodity))

## [1] 553

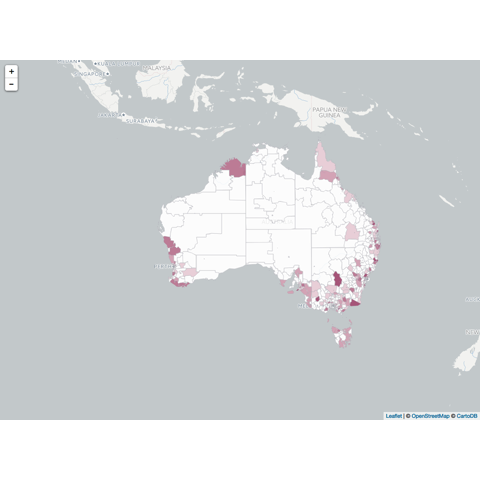
#If you want to see what they are, run this  
 #sort(unique(agri\_regions$Commodity))  
  
 #Flowers  
 flowers\_regions <-filter(agri\_regions, agri\_regions$Commodity ==   
 "Nurseries cut flowers and cultivated turf - Total area (ha)")  
 flowers\_regions<-select(flowers\_regions, c(1,3))  
 flowers\_regions<-ddply(flowers\_regions,.(SA2\_NAME11),summarize,Total\_Area\_Flowers=sum(Value,   
 na.rm = TRUE))  
  
 #merge  
 Aust<-merge(Aust, flowers\_regions, by="SA2\_NAME11")  
  
 #Broadacre  
 broadacre\_regions <-filter(agri\_regions, agri\_regions$Commodity ==   
 "Broadacre crops - Cereal crops - Barley for grain - Area (ha)" | agri\_regions$Commodity ==   
 "Broadacre crops - Cereal crops - Wheat for grain - Area (ha)")  
 broadacre\_regions<-select(broadacre\_regions, c(1,3))  
 broadacre\_regions<-ddply(broadacre\_regions,.(SA2\_NAME11),summarize,Total\_Area\_Crops=sum(Value, na.rm = TRUE))  
  
 #merge  
 Aust<-merge(Aust, broadacre\_regions, by="SA2\_NAME11")  
  
 #Fruit and nuts  
 a<-"Fruit and nuts - Berry fruit - Total area (ha)"  
 b<-"Fruit and nuts - Orchard fruit and nut trees - Total area (ha)"  
 c<-"Fruit and nuts - Other fruit - All other fruit - Total area (ha)"   
 d<-"Grapevines - Grapevines for wine production - Total Area (ha)"  
  
 fruit\_regions <-filter(agri\_regions, agri\_regions$Commodity == a | agri\_regions$Commodity ==   
 b | agri\_regions$Commodity == c | agri\_regions$Commodity == d)  
 fruit\_regions<-select(fruit\_regions, c(1,3))  
 fruit\_regions<-ddply(fruit\_regions,.(SA2\_NAME11),summarize,Total\_Area\_Fruit=sum(Value, na.rm = TRUE))  
  
 #merge  
 Aust<-merge(Aust, fruit\_regions, by="SA2\_NAME11")  
  
 #Hay  
 hay\_regions <-filter(agri\_regions, agri\_regions$Commodity == "Hay and Silage - Hay - Total area (ha)")  
 hay\_regions <-select(hay\_regions, c(1,3))   
 hay\_regions <-ddply(hay\_regions,.(SA2\_NAME11),summarize,Total\_Area\_Hay=sum(Value, na.rm = TRUE))  
  
 #merge  
 Aust<-merge(Aust, hay\_regions, by="SA2\_NAME11")  
  
 #Livestock  
 livestock\_regions <-filter(agri\_regions,agri\_regions$Commodity == "Livestock - Cattle - Total (no.)")  
 livestock\_regions <-select(livestock\_regions, c(1,3))   
 livestock\_regions <-ddply(livestock\_regions,.(SA2\_NAME11),summarize,Total\_Area\_Livestock=sum(Value, na.rm = TRUE))  
  
 #merge  
 Aust<-merge(Aust, livestock\_regions, by="SA2\_NAME11")  
  
 #Vegetables  
 veg\_regions <- filter(agri\_regions, agri\_regions$Commodity ==   
 "Vegetables for human consumption - All other - Area (ha)" )  
 veg\_regions <- select(veg\_regions,c(1,3))   
 veg\_regions <- ddply(veg\_regions,.(SA2\_NAME11),summarize,Total\_Area\_Veg=sum(Value, na.rm = TRUE))   
  
 #merge  
 Aust<-merge(Aust, veg\_regions, by="SA2\_NAME11")  
  
 #replace NA with 0   
 Aust@data[is.na(Aust@data)]<-0  
 #save(Aust, file = "Aust\_Pop\_Industry.Rdata")

Make maps showing various industries.

#plot flowers  
pal\_flowers<- colorBin(c("white","deeppink4"), domain= Aust$Total\_Area\_Flowers)  
map\_flowers<-leaflet(data = spTransform(Aust,CRS(proj))) %>% addProviderTiles("CartoDB.Positron") %>%  
 addPolygons(fillColor = ~pal\_flowers(Total\_Area\_Flowers), # refers to pal defined above  
 fillOpacity = 0.8,   
 color = "#BDBDC3", # colour between polygons  
 weight = 1,  
 highlight = highlightOptions(  
 weight = 5,  
 color = "pink",  
 dashArray = "",  
 fillOpacity = 0.7,  
 bringToFront = TRUE))  
  
mapshot(map\_flowers, file = "Map\_Flowers.png")  
  
#plot broadacre crops  
pal\_crops<- colorBin(c("white","peru"), domain= Aust$Total\_Area\_Crops)  
map\_flowers<-leaflet(data = spTransform(Aust,CRS(proj))) %>% addProviderTiles("CartoDB.Positron") %>%  
 addPolygons(fillColor = ~pal\_crops(Total\_Area\_Crops), # refers to pal defined above  
 fillOpacity = 0.8,   
 color = "#BDBDC3", # colour between polygons  
 weight = 1,  
 highlight = highlightOptions(  
 weight = 5,  
 color = "pink",  
 dashArray = "",  
 fillOpacity = 0.7,  
 bringToFront = TRUE))  
  
mapshot(map\_flowers, file = "Map\_Crops.png")  
  
#plot fruit  
pal\_fruit<- colorBin(c("white","peru"), domain= Aust$Total\_Area\_Fruit)  
map\_flowers<-leaflet(data = spTransform(Aust,CRS(proj))) %>% addProviderTiles("CartoDB.Positron") %>%  
 addPolygons(fillColor = ~pal\_fruit(Total\_Area\_Fruit), # refers to pal defined above  
 fillOpacity = 0.8,   
 color = "#BDBDC3", # colour between polygons  
 weight = 1,  
 highlight = highlightOptions(  
 weight = 5,  
 color = "pink",  
 dashArray = "",  
 fillOpacity = 0.7,  
 bringToFront = TRUE))  
  
mapshot(map\_flowers, file = "Map\_Fruit.png")  
  
#plot veg  
pal\_veg<- colorBin(c("white","green4"), domain= Aust$Total\_Area\_Veg)  
map\_flowers<-leaflet(data = spTransform(Aust,CRS(proj))) %>% addProviderTiles("CartoDB.Positron") %>%  
 addPolygons(fillColor = ~pal\_veg(Total\_Area\_Veg), # refers to pal defined above  
 fillOpacity = 0.8,   
 color = "#BDBDC3", # colour between polygons  
 weight = 1,  
 highlight = highlightOptions(  
 weight = 5,  
 color = "pink",  
 dashArray = "",  
 fillOpacity = 0.7,  
 bringToFront = TRUE))  
  
mapshot(map\_flowers, file = "Map\_Veg.png")  
  
#plot hay  
pal\_hay<- colorBin(c("white","gold3"), domain= Aust$Total\_Area\_Hay)  
map\_flowers<-leaflet(data = spTransform(Aust,CRS(proj))) %>% addProviderTiles("CartoDB.Positron") %>%  
 addPolygons(fillColor = ~pal\_hay(Total\_Area\_Hay), # refers to pal defined above  
 fillOpacity = 0.8,   
 color = "#BDBDC3", # colour between polygons  
 weight = 1,  
 highlight = highlightOptions(  
 weight = 5,  
 color = "pink",  
 dashArray = "",  
 fillOpacity = 0.7,  
 bringToFront = TRUE))  
  
mapshot(map\_flowers, file = "Map\_Hay.png")  
  
#plot livestock  
pal\_livestock<- colorBin(c("white","lavenderblush4"), domain= Aust$Total\_Area\_Livestock)  
map\_flowers<-leaflet(data = spTransform(Aust,CRS(proj))) %>% addProviderTiles("CartoDB.Positron") %>%  
 addPolygons(fillColor = ~pal\_livestock(Total\_Area\_Livestock), # refers to pal defined above  
 fillOpacity = 0.8,   
 color = "#BDBDC3", # colour between polygons  
 weight = 1,  
 highlight = highlightOptions(  
 weight = 5,  
 color = "pink",  
 dashArray = "",  
 fillOpacity = 0.7,  
 bringToFront = TRUE))  
  
mapshot(map\_flowers, file = "Map\_Livestock.png")

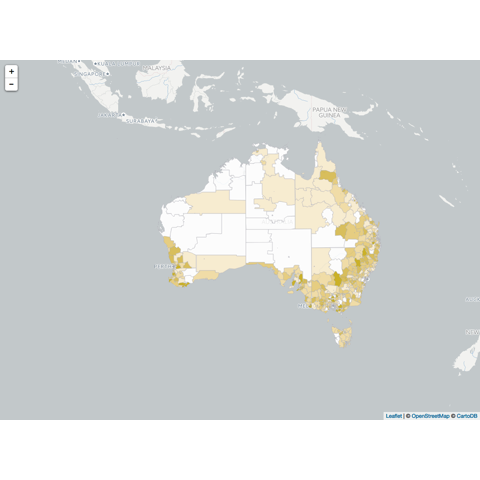
Show map of flower cultivation and hay production areas, as examples.

img <- readPNG("Map\_Flowers.png")  
 grid.raster(img)



While hay is produced here:

img <- readPNG("Map\_Hay.png")  
 grid.raster(img)



## Simulating maximum range of invasion for 20 pests

The proper way to do this is to use CLIMEX model or something similar, because CEBRA application is an exploratory tool minimally plausible distributions are sufficient. We will use the SA2 data to model these ranges. The species considered come from [Australian Government, Department of Agriculture](http://www.agriculture.gov.au/pests-diseases-weeds/plant/xylella).

Looking at the industry data makes it unclear what the units are, it seems unlikely that these are indeed in "ha" as it would appear that more area is cultivated than is available. That's not a problem though for this coarse excercise as we will just treat it as absence/presense in an SA2 unit information.

species<-c("Xylella","Khapra beetle",  
 "Exotic fruit fly","Karnal bunt","Huanglongbing","Gypsy moths",  
 "Tramp ants","Bees mites","Giant African snail","Stink bug",  
 "Zebra chip","Ug99","Russian wheat aphid","Citrus canker",  
 "Guava rust","Airborne phytophthora","Exotic bees",  
 "Panama disease tropical race 4","Potato cyst nematode","Leaf miner")  
   
 #Select commodity key words to approximate the range of respective invasive species  
 host<- c("Grapevines", "seed", "Fruit", "Wheat", "Citrus", "Total", "and","Orchard",  
 "Vegetables","flower","Tomatoes", "Barley","Broadacre", "Citrus","Trees","Trees","rchard","Bananas",  
 "Potatoes","t/ha")  
   
   
 species\_host<-data.frame(Species=species, Selected\_by= host)  
 species\_host

## Species Selected\_by  
## 1 Xylella Grapevines  
## 2 Khapra beetle seed  
## 3 Exotic fruit fly Fruit  
## 4 Karnal bunt Wheat  
## 5 Huanglongbing Citrus  
## 6 Gypsy moths Total  
## 7 Tramp ants and  
## 8 Bees mites Orchard  
## 9 Giant African snail Vegetables  
## 10 Stink bug flower  
## 11 Zebra chip Tomatoes  
## 12 Ug99 Barley  
## 13 Russian wheat aphid Broadacre  
## 14 Citrus canker Citrus  
## 15 Guava rust Trees  
## 16 Airborne phytophthora Trees  
## 17 Exotic bees rchard  
## 18 Panama disease tropical race 4 Bananas  
## 19 Potato cyst nematode Potatoes  
## 20 Leaf miner t/ha

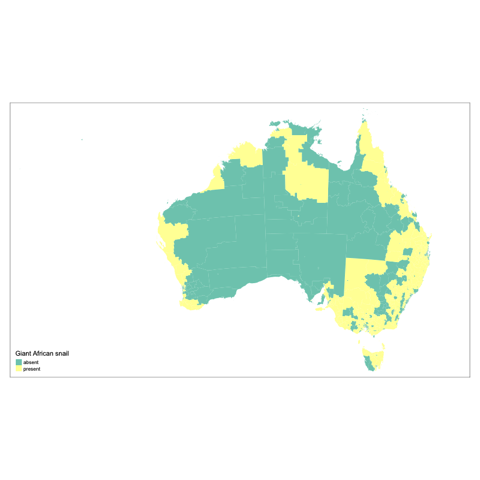
write.csv(species\_host, "Species\_Host.csv")   
   
 #Simulate species distributions as absence/presence values by region  
 #And merge with Aust  
 #grepl function lets us choose all the regions that have to do with a specific host  
 for (i in 1:length(species))  
 {  
 temp<- filter(agri\_regions, grepl(host[i], agri\_regions$Commodity))  
 temp <- select(temp,c(1,3))   
 temp <- ddply(temp,.(SA2\_NAME11),summarize, Indicator =sum(Value, na.rm = TRUE))  
 temp$Indicator[is.na(temp$Indicator)]<-"absent"  
 temp$Indicator[temp$Indicator>0]<-"present"  
 names(temp)<-c("SA2\_NAME11", species[i])  
 Aust<-merge(Aust, temp, by="SA2\_NAME11")  
   
 }  
  
 #replace NA with "absent"  
 Aust@data[is.na(Aust@data)]<-"absent"  
 #save(Aust, file = "Australia.Rdata")

Now we can make maps that show simulated maximum spread of these invasive pests.

for (i in 1:length(species))  
 {  
 #file = name of the destinatin file  
 file<-paste("www/",species[i],"\_map",".png", sep="")  
 png(file = file, bg = "white", width = 800, height=500)  
 par(mar=c(0,0,0,0))  
 print(qtm(shp = Aust, fill = species[i],borders = NULL))  
 dev.off()  
 }

An example of a pest ivasion map.

img <- readPNG("www/Giant African snail\_map.png")  
 grid.raster(img)



# Prepare inputs for the CEBRA app

Now we can make a table that will provide inputs for the app. We will record the population each pest affects, area of invasion, area of overlap with each each industry and main population centers (above certain density), proportion of value at risk for each industry, which is calculated by taking the [area where industry and pests overlap]/[total area of industry].

#Probably should not be making extra copies of data without need  
 x<-Aust@data  
 #head(x)  
  
 input\_app<-matrix(NA,nrow = length(species), ncol = 7)  
   
 for (i in 1:length(species))  
 {  
 y<-subset(x, x[,species[i]]=="present" & x$Total\_Area\_Flowers > 0)  
 input\_app[i,1]<-sum(y$Area\_ha)  
  
 y<-subset(x, x[,species[i]]=="present" & x$Total\_Area\_Crops > 0)  
 input\_app[i,2]<-sum(y$Area\_ha)  
  
 y<-subset(x, x[,species[i]]=="present" & x$Total\_Area\_Fruit > 0)  
 input\_app[i,3]<-sum(y$Area\_ha)  
  
 y<-subset(x, x[,species[i]]=="present" & x$Total\_Area\_Hay > 0)  
 input\_app[i,4]<-sum(y$Area\_ha)  
  
 y<-subset(x, x[,species[i]]=="present" & x$Total\_Area\_Livestock > 0)  
 input\_app[i,5]<-sum(y$Area\_ha)  
  
 y<-subset(x, x[,species[i]]=="present" & x$Total\_Area\_Veg > 0)  
 input\_app[i,6]<-sum(y$Area\_ha)  
  
 y<-subset(x, x[,species[i]]=="present")  
 input\_app[i,7]<-sum(y$Population)  
 }  
 #tidy the dataframe  
 colnames(input\_app)<-c("Overlap\_Flowers","Overlap\_Crops",  
 "Overlap\_Fruit","Overlap\_Hay","Overlap\_Livestock","Overlap\_Veg","Overlap\_Population\_Numbers")  
 input\_app<-data.frame(input\_app)  
 input\_app<-mutate(input\_app, Species=species)  
 input\_app<-input\_app[,c(8,1:7)]  
   
 y<-subset(x, x$Total\_Area\_Flowers > 0)  
 Total\_Flowers<-sum(y$Area\_ha)  
  
 y<-subset(x, x$Total\_Area\_Crops > 0)  
 Total\_Crops<-sum(y$Area\_ha)  
  
 y<-subset(x, x$Total\_Area\_Fruit > 0)  
 Total\_Fruit<-sum(y$Area\_ha)  
  
 y<-subset(x, x$Total\_Area\_Hay > 0)  
 Total\_Hay<-sum(y$Area\_ha)  
  
 y<-subset(x, x$Total\_Area\_Livestock > 0)  
 Total\_Livestock<-sum(y$Area\_ha)  
  
 y<-subset(x, x$Total\_Area\_Veg > 0)  
 Total\_Veg <-sum(y$Area\_ha)  
  
 Total\_Population <-sum(x$Population)  
  
 input\_app<-mutate(input\_app, Prop\_Flowers = Overlap\_Flowers/Total\_Flowers)  
 input\_app<-mutate(input\_app, Prop\_Fruit = Overlap\_Fruit/Total\_Fruit)  
 input\_app<-mutate(input\_app, Prop\_Veg = Overlap\_Veg/Total\_Veg)  
 input\_app<-mutate(input\_app, Prop\_Hay = Overlap\_Hay/Total\_Hay)  
 input\_app<-mutate(input\_app, Prop\_Crops = Overlap\_Crops/Total\_Crops)  
 input\_app<-mutate(input\_app, Prop\_Livestock = Overlap\_Livestock/Total\_Livestock)  
 input\_app<-mutate(input\_app, Prop\_Population = Overlap\_Population\_Numbers/Total\_Population)

## Include economic data

The data on economic value comes from [Australian Bureau of Statistics](http://www.abs.gov.au/ausstats/abs@.nsf/0/58529ACD49B5ECE0CA2577A000154456?Opendocument).

'VALUE OF AGRICULTURAL COMMODITIES PRODUCED, Australia, year ended 30 June 2016', in aggregate, are used and added to the inputs.

#Values from the website linked above, in millions of Australian dollars  
 Value\_Flowers<-1296.3  
 #Wheat, Oates,Barley, Sorghum,and Rice   
 Value\_Crops<-6170.1+397.9+2276.6+491.6+114.8  
 #Fruit and grapes  
 Value\_Fruit<-4224.6+1333.9  
 #http://www.abs.gov.au/ausstats/abs@.nsf/Lookup/by%20Subject/1301.0~2012~Main%20Features~Agricultural%20production~260  
 Value\_Hay<-1600  
 #Slaughtering and products  
 Value\_Livestock<-20622.5+8029.9  
   
 Value\_Veg <-3585.4  
   
 input\_app<-mutate(input\_app, Econ\_Flowers= Prop\_Flowers\*Value\_Flowers)  
 input\_app<-mutate(input\_app, Econ\_Fruit =Prop\_Fruit\*Value\_Fruit)  
 input\_app<-mutate(input\_app, Econ\_Veg = Prop\_Veg\*Value\_Veg)  
 input\_app<-mutate(input\_app, Econ\_Hay =Prop\_Hay\*Value\_Hay)  
 input\_app<-mutate(input\_app, Econ\_Crops =Prop\_Crops\*Value\_Crops)  
 input\_app<-mutate(input\_app, Econ\_Livestock = Prop\_Livestock\*Value\_Livestock)

## Include default values that can be altered in the app

By looking at the proportion of overlap between species distribution and key industries, we have arrived at values at risk. However, it is unlikely that the pest will cause a 100% damage whenever it comes in contact with a commodity. In the app the extent of damage for each species and each industry can be modified.

Read default values from an 'Impact\_Species\_Host' file.

Store "Input\_App.csv" in the 'data' directory for use by the app, this file now contains default values for monetary and non-monetary damages, management utility, time and unweighted impact scores that produce default ranking.

file<-"Impact\_Species\_Host.xls"  
 #Read the file  
 impacts <- read\_excel(file, sheet="Impacts", skip=0)  
   
 input\_app<-merge(input\_app, impacts, by="Species")  
   
 #Add a column with total monetary damage  
 input\_app<-mutate(input\_app, Monetary\_Damage= (Econ\_Flowers\*Impact\_Flowers+  
 Econ\_Fruit\*Impact\_Fruit+  
 Econ\_Veg\*Impact\_Veg+  
 Econ\_Hay\*Impact\_Hay+  
 Econ\_Crops\*Impact\_Crops+  
 Econ\_Livestock\*Impact\_Livestock))  
   
 input\_app<-mutate(input\_app, Man\_Utility = 4.9)  
   
 input\_app<-mutate(input\_app, Unweighted\_Score = (Monetary\_Damage/max(Monetary\_Damage)+  
 Amenity/max(Amenity)+  
 Environment/max(Environment)+   
 Man\_Utility/max(Man\_Utility)+  
 (1-Time/max(Time))))   
   
 #Generate random likelihood of entry, spread and establishment   
 set.seed(347)  
 chance<-rnorm(20,0.6, 0.2)  
 social<-round(rnorm(20,3, 1))+1  
   
 input\_app<-mutate(input\_app, Likelihood = chance)   
 #Generate random community impact scores  
 input\_app<-mutate(input\_app, Community = social)   
   
 write.csv(input\_app, "data/Input\_App.csv")   
   
 #Look at the names of the top 10 largest by utility  
 input\_app[order(-input\_app$Unweighted\_Score),][1:10,][,c("Species", "Monetary\_Damage", "Unweighted\_Score")]

## Species Monetary\_Damage Unweighted\_Score  
## 17 Tramp ants 7879.6000 4.700000  
## 12 Leaf miner 4458.5432 4.065834  
## 10 Karnal bunt 4247.5186 3.439053  
## 11 Khapra beetle 1890.2000 3.339885  
## 5 Exotic fruit fly 1659.3214 3.310584  
## 4 Exotic bees 2513.0604 3.218932  
## 2 Bees mites 551.3355 3.169970  
## 8 Gypsy moths 5014.3600 3.136372  
## 9 Huanglongbing 1434.0852 3.082000  
## 15 Russian wheat aphid 6044.5495 3.067114

#Look at the names of the top 10 largest by monetary damage  
 input\_app[order(-input\_app$Monetary\_Damage),][1:10,][,c("Species", "Monetary\_Damage", "Unweighted\_Score")]

## Species Monetary\_Damage Unweighted\_Score  
## 17 Tramp ants 7879.600 4.700000  
## 15 Russian wheat aphid 6044.550 3.067114  
## 8 Gypsy moths 5014.360 3.136372  
## 12 Leaf miner 4458.543 4.065834  
## 10 Karnal bunt 4247.519 3.439053  
## 4 Exotic bees 2513.060 3.218932  
## 11 Khapra beetle 1890.200 3.339885  
## 18 Ug99 1872.849 2.537683  
## 5 Exotic fruit fly 1659.321 3.310584  
## 9 Huanglongbing 1434.085 3.082000

## Alternative visualisations

In addition to ranking, it is possible to visualise relative risks in two dimensions: impact uncertainty (randomness comes from uncertainty in entry, spread, and establishment) and management uncertainty (uncertainty distribution is derived from expert beliefs about possibilities of eradication).

theme\_set(theme\_classic())  
 mon\_damage<-input\_app$Monetary\_Damage[input\_app$Species=="Xylella"]  
 risk\_dist<-mon\_damage\*rnorm(10000,mean=input\_app$Likelihood[input\_app$Species=="Xylella"], sd = 0.1)  
 management\_unc<-rnorm(10000, 0.6, 0.05)  
  
 df<-data.frame(score=risk\_dist,management=management\_unc, species="Xylella")  
  
 mon\_damage<-input\_app$Monetary\_Damage[input\_app$Species=="Ug99"]  
 risk\_dist<-mon\_damage\*rnorm(10000,mean=input\_app$Likelihood[input\_app$Species=="Ug99"], sd = 0.1)  
 management\_unc<-rnorm(10000, 0.4, 0.1)  
  
 df <- rbind(df, data.frame(score=risk\_dist,management=management\_unc, species="Ug99"))  
  
 p <- ggplot(data=df, aes(x=score, y=management,colour=species)) + geom\_point(size=0.5, alpha=.6) +  
 stat\_ellipse(data=df, aes(x=score, y=management,colour=species, fill=factor(df$species)), size=1,  
 geom="polygon",level=0.95,alpha=0.4)+  
 xlab("Monetary damage uncertainty")+ylab("Management uncertainty")+  
 scale\_fill\_manual(values=c("orange","blue"), name="95% density",   
 labels=unique(df$species))  
 p

