STAT 585X - Final Project - Report
Changes in cultivated cropland in CEAP

Andreea Erciulescu

April 25, 2014

1 Introduction

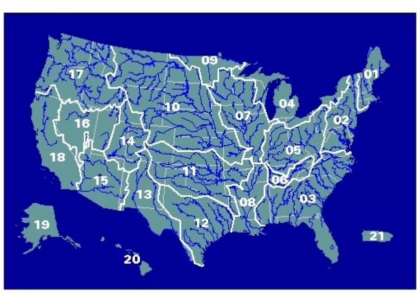
1.1 Question of interest

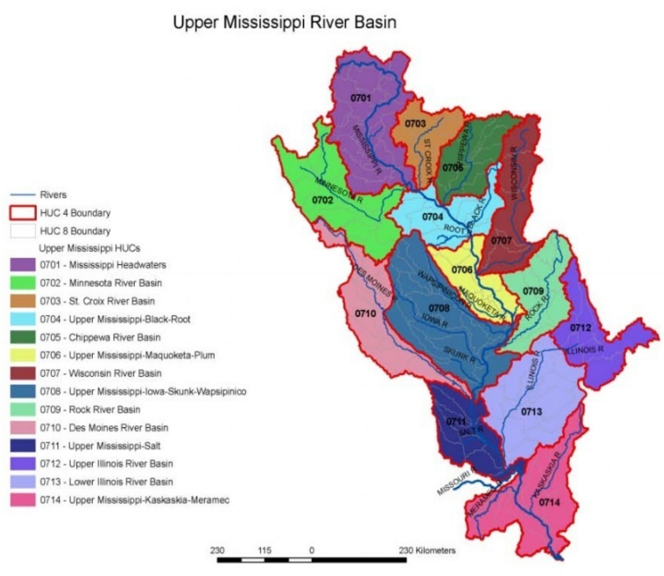
Do changes in land characteristics over time match the Conservation Effects Assessement Project (CEAP) survey design and data collection?

1.2 Motivation

National Resources Inventory (NRI) is an annual survey conducted collaboratively by USDA NRCS (Natural Resources Conservation Services) and ISU Center for Survey Statistics and Methodology (CSSM) to provide status and trend estimates for natural resources on nonfederal lands in US. Example of such estimates are soil erosion estimates in relation to land characteristics and programs.

Conservation Effects Assessment Project (CEAP) is a series of surveys intended to quantify environmental effects of conservation prctices and programs by hydrologic unit codes (HUCs). The following image illustrates the division of the United States territory into 2-digits HUCs and the sudivision of the Upper Mississippi region into 4-digit HUCs.





The CEAP sample is a subset of the NRI points classified as cultivated cropland. The data was collected using farmers interviews and NRCS hydrologic, climate and soil databases. The data was then filtered through an APEX model (black box) and the result is a set of observations for usable points for 19 response variables. The final goal is to produce erosion estimates for 8-digit HUCs.

CEAP is run at both national and regional levels. The Des Moines River Watershed (HUC 0710) is the region of interest in this project, that is subdivided into nine 8-digit HUCs.

For CSSM analysis, data from the Soil Survey is considered, collected over the 2003 to 2006 time period. Hence, the survey frame, consisting of NRI cultivated cropland points, is set in 2003. The information for these eligible points is collected in the following years, 2004-2007. Changes in land characteristics for the sample points may result in frame coverage problems that should not be ignored in the analysis.

In this project we are using publicly available land data to investigate changes in land characteristics over time for a region of interest in CEAP.

1.3 Data

The CEAP sample data for the Des Moines River Watershed (HUC 0710) region is not publicly available. Web navigation brings us to different sources of data on the United States counties and regions. In this project, we consider the following sources of information:

- Crop Data Layer (CDL)

  The CDL data is available at `http://nassgeodata.gmu.edu/CropScape/` in the form of Tagged Image File (.tif) Format. We are interested in the state of Iowa data, available for the years of 2003-2007. The information consists of pixel counts and acreage values for different categories of cropland data. Each of the category has an associated value (code), for example 1 stands for Corn and 5 stands for Soybean. A complete list of category codes and class names for the USDA NASS CDL is available at `http://www.nass.usda.gov/research/Cropland/docs/CDL_2013_crosswalk.htm`.

- Census Topologically Integrated Geographic Encoding and Referencing database (Tigerweb)

  The Census Tigerweb data is available at `http://tigerweb.geo.census.gov/tigerwebmain` for both national and regional levels. Also, data for the hydrologic levels is available at

  `http://tigerweb.geo.census.gov/tigerwebmain/Files/tigerweb_tab10_hydro_poly_ia.html`. We are interested in the state of Iowa data, as well as the Des Moines River data. In particular, we are interested in the points coordinates.

- Public Land Survey System (PLSS)

  The PLSS data can be found on `http://www.geocommunicator.gov/GeoComm/lsis_home/home/index.htm` in the form of shapefiles. Information is available at both state and county levels.

- GIS data on hydrologic basins

  The GIS data can be found at `ftp://ftp.igsb.uiowa.edu/gis_library/basins/` in the form of shapefiles. Information is available for the entire Des Moines River basin.

- Atlas of historical countyu boundaries (AtlasHCB)

  The AtlasHCB data is available at `http://publications.newberry.org/ahcbp/pages/Iowa.html` in the form of shapefiles. Information is available for the entire state of Iowa.

2 Data Collection and Processing Steps

We explored all these different data sources using R tools. These data differ in format, in dimension and, most importantly, in the enclosed information. We are interested in a very specific region in the state of Iowa so it is very important to select the sources that provide us with most useful information.

Except for the CENSUS Tigerweb data and the CDL codes and classification data, all other files need to be downloaded and stored in the working folder. If you are not able to download it from the web (steps are presented below), please contact us and we will provide you the data.

2.1 CDL data

We first download the data for years 2003-2007 from the website, following these steps:

1. Open this link in an internet browser: `http://nassgeodata.gmu.edu/CropScape/`. We now have the United States map and some tabs on the left and at the top of the map;
2. Click on the *US map -like* tab at the top, hoovering over it says *Define Area of Interest by Region/State/ASD/County*, select *State* as the *Level* and *Iowa* as the *state*. Click *Submit*;
3. Select the years of interest (in this case 2003-2007) and download the data. The file is large, so downloading it takes long time.

At the end of the downloading process, we unzip the file and we have five .tif files, one for each year of interest. These are raster graphics images, spatial data structures that divide the US teritory into pixels that store crop information. This type of data is referred to as a 'grid,' contrasted with 'vector' data that is used to represent points, lines, polygons. The dimensions of the files are large, about 200,000 KB each for the 2003-2005 data and about 60,000 KB each for the 2006-2007 data.

Note: The first and greatest challenge in this section was storing the CDL data. We run our of memory on the working drive (U drive) and had to consider document reallocation on web storage clouds.

An useful R package to read and manipulate the CDL data is the *Raster* package, implemented using S4 methods. This package allows us to read the raster values from the files and to convert the cell numbers to coordinates and back.

```
## read the data
cdl.ia03 <- raster("data/CDL_2003_19.tif")
cdl.ia04 <- raster("data/CDL_2004_19.tif")
cdl.ia05 <- raster("data/CDL_2005_19.tif")
cdl.ia06 <- raster("data/CDL_2006_19.tif")
cdl.ia07 <- raster("data/CDL_2007_19.tif")

cdl.ia03

## class       : RasterLayer
## dimensions  : 11672, 17796, 207714912  (nrow, ncol, ncell)
## resolution  : 30, 30  (x, y)
## extent      : -52065, 481815, 1938165, 2288325  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=23 +lon_0=-96 +x_0=0 +y_0=0 +ellps=GRS80
## data source : U:\stat585\STAT585X-Project\data\CDL_2003_19.tif
## names       : CDL_2003_19
## values      : 0, 255  (min, max)
```

Notice the attributes of the raster objects. One of the most important attributes, for us, is the coordinate reference system (CRS), or the map projection. We need to pay greater attention to this attribute in order to carefully manipulate the future data on the region of interest, to get the matching coordinate reference (more details follow). The objects we have so far are 'skeletons,' because they only contain attributes of the data, not the actual values stored. In other words, *cdl.ia03 -like* objects are *RatserLayers* that do not contain any cell (pixel) value in (RAM) memory, they only contain the parameters that describe the *RasterLayers*. We are going to read the values for the region of interest using cell numbers and coordinates (xy) in the extraction method using the *cellFromXY* function. For this, we need the coordinates for the Des Moines River Watershed and surrounding area, denoted as the region of interest.

2.2 GIS, PLSS, AtlasHCB data

GIS, PLSS, AtlasHCB data sources are considered at first. However, we do not use any in the final results. The dimensions are smaller than the dimensions of the CDL data, so downloading and storing is not as demanding anymore. However, the information in the data is not realible and, as it turns out, not useful for our purposes.

We first download the data from the websites. We read in the data and we extract the polygons information from the shapefiles using the *maptools* library in R and the function developed in one of the STAT 585X labs, to extract the county based information.

For the GIS data, after mapping the region, we realized that it covers the entire basin, more than what we are interested in. Also, the polygon data is in the universal transverse mercador (UTM) and we need to convert it to the longitude-latitude, then to the CRS with the appropriate raster characteristics.

The information in the AtlasHCB data is based on historical records and it is only county based. Hence it is not useful for our purposes because the time period (2003-2007) and the specific region (Des Moines River Watershed) are not included in the specifics of AtlasHCB data.

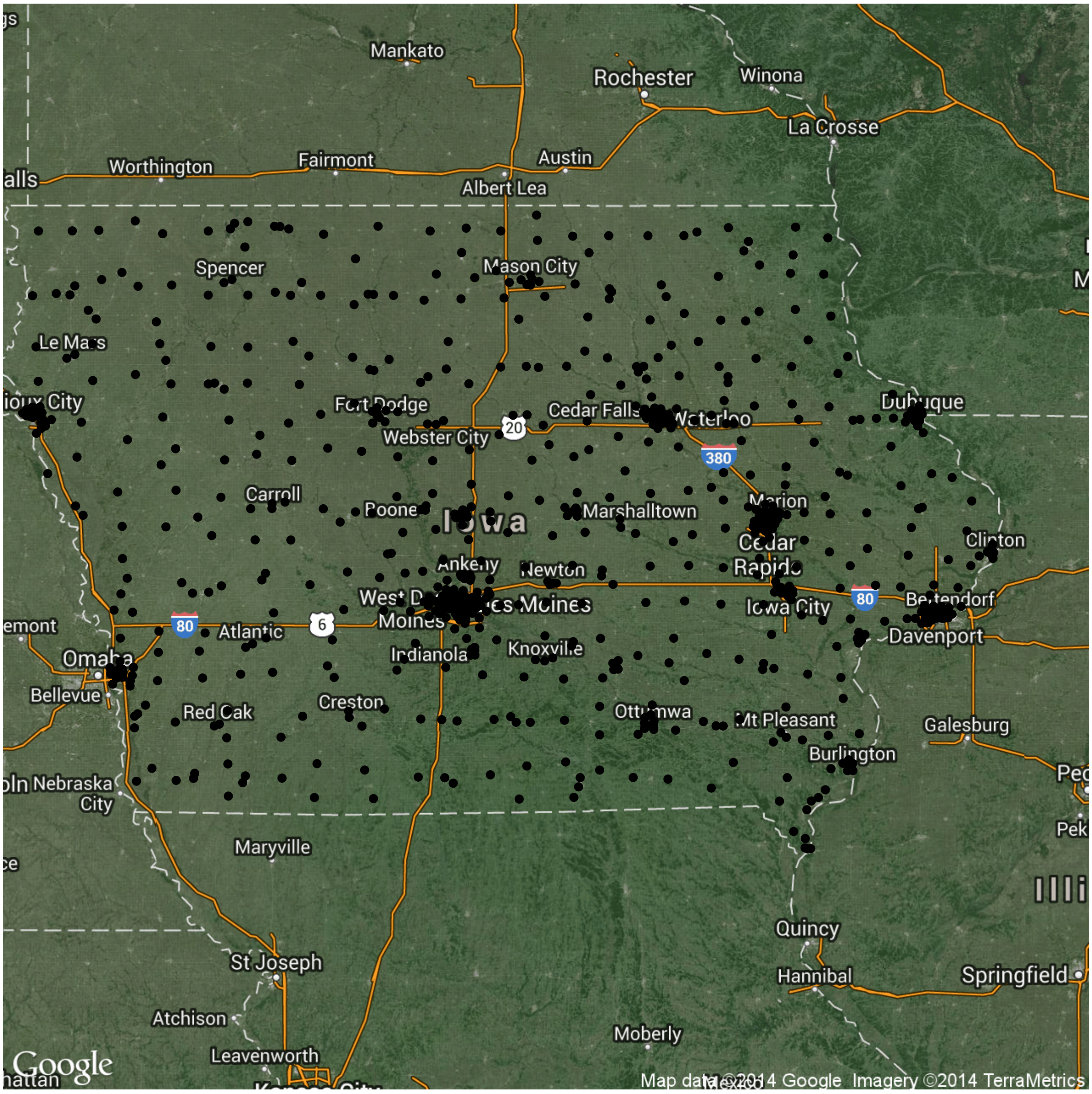
On the other hand, PLSS data have large dimensions and storage space has been a challenge in this project. The processing steps are similar to the ones described above, in this section, because we are once again dealing with shapefiles. Also, the data is available at both state and county levels, but not at hydrologic levels. Hence, we decide to search for another source of data. Census data turns out to be very useful for us, and we decribe it in the next section.

2.3 Census data

We pull both the Iowa data and the Des Moines River data from the web using the *XML* library in R. The Iowa data is used to extract the point coordinates and construct a map. First, we pull the point coordinates for the available points and save them in a dataframe containing the numeric values. We use the *ggplot2* library in R to construct a plot of the data and the *maps* and *ggmap* libraries to construct a map of the data. These visualisations serve as both a nice representation of the data and as a verification that we are working with the desired region. We constructed maps using both *qplot* and *qmap* functions but we only present the google maps.

```
### pull the iowa census data from the web
src <- "http://tigerweb.geo.census.gov/tigerwebmain/Files/tigerweb_acs13_tract_ia.html"
tables <- readHTMLTable(src)
ldply(tables, dim)
```

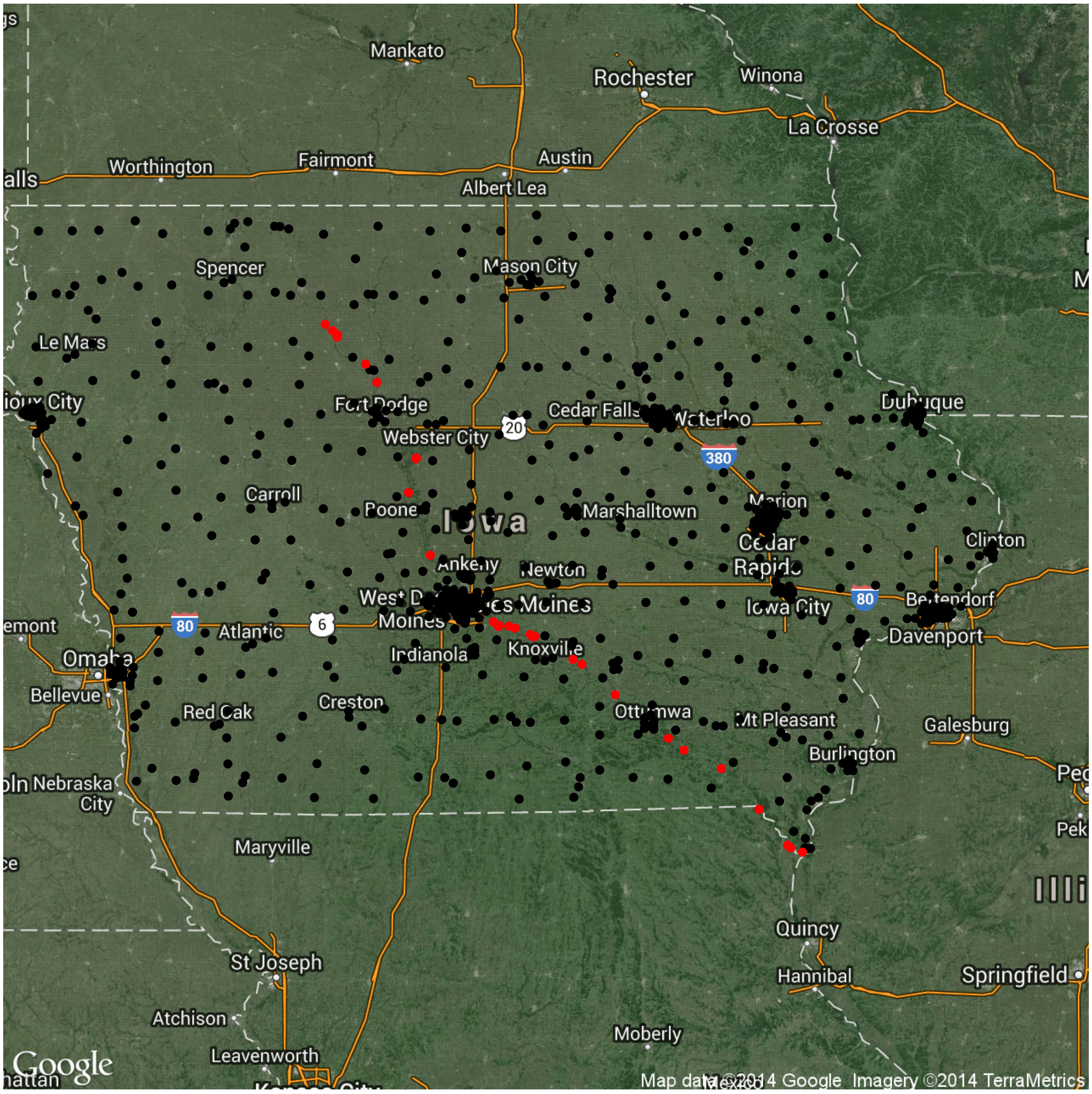
```
##                                                                         .id  V1 V2
## 1 State of Iowa Census Tracts - Current/ACS13 - Data as of January 1, 2013 825 17

tigeria <- tables[[1]]
head(tigeria)

##   MTFCC            OID       GEOID STATE COUNTY  TRACT BASENAME              NAME LSADC
## 1 G5020 20790328763043 19001960100    19    001 960100     9601 Census Tract 9601    CT
## 2 G5020 20790328763062 19001960200    19    001 960200     9602 Census Tract 9602    CT
## 3 G5020 20790328763079 19001960300    19    001 960300     9603 Census Tract 9603    CT
## 4 G5020 20790155900148 19003950100    19    003 950100     9501 Census Tract 9501    CT
## 5 G5020 20790155900144 19003950200    19    003 950200     9502 Census Tract 9502    CT
## 6 G5020 20790692090375 19005960100    19    005 960100     9601 Census Tract 9601    CT
##   FUNCSTAT  AREALAND AREAWATER UR     CENTLAT      CENTLON    INTPTLAT     INTPTLON
## 1        S 707919439   1124009  ÃĆ +41.4204490 -094.4773809 +41.4226701 -094.4754745
## 2        S 666068268   1098841  ÃĆ +41.2402687 -094.4593481 +41.2403358 -094.4634803
## 3        S 100416467    375146  ÃĆ +41.2975066 -094.5025203 +41.2964752 -094.5071226
## 4        S 879308484   4856978  ÃĆ +41.0477119 -094.6732029 +41.0509146 -094.6680804
## 5        S 217392267    496426  ÃĆ +40.9528920 -094.8044597 +40.9482672 -094.8079792
## 6        S 335670054  27231318  ÃĆ +43.3462135 -091.2527640 +43.3464049 -091.2647941

### transform the data to the desired format

## first we need to pull the coordinates
dsmriv <- tigeria
poly.coords.ia <- as.data.frame(dsmriv[, c("INTPTLAT", "INTPTLON")])
row.names(poly.coords.ia) <- NULL
names(poly.coords.ia) <- c("x", "y")

poly.coords.ia[, 1] <- as.numeric(as.vector(poly.coords.ia[, 1]))
poly.coords.ia[, 2] <- as.numeric(as.vector(poly.coords.ia[, 2]))

## plot the area qplot(y,x,data=poly.coords.ia)

qmap(location = "iowa", zoom = 7, maptype = "hybrid") + geom_point(data = poly.coords.ia, mapping =
    y = x), size = 2)
### always remember, longitude first, then latitude!!!!!!
```



The data for the Des Moines River is extracted from the full dataframe on the hydrologic levels in Iowa. Again, we construct a dataframe with numeric values for the point coordinates and we overlay this data on the previous map.

```
### pull the hydrologic iowa data from the web

src <- "http://tigerweb.geo.census.gov/tigerwebmain/Files/tigerweb_tab10_hydro_poly_ia.html"
tables <- readHTMLTable(src)
ldply(tables, dim)

##                                                                         .id    V1 V2
## 1 U.S. Polygon Hydrography for Iowa - Data as of January 1, 2010 16448 15

tigeria <- tables[[1]]
head(tigeria)
```

```
##   MTFCC          OID PRETYP PRETYPEABRV SUFTYP SUFTYPEABRV                 BASENAME
## 1 H3010 110697284649      ÃĆ            ÃĆ      ÃĆ            ÃĆ             Ackerman Cut
## 2 H3010 110420222385      ÃĆ            ÃĆ    225         Crk                     Allen
## 3 H2030 110438005187      ÃĆ            ÃĆ    361          Lk                 Alligator
## 4 H2030 110417623043      ÃĆ            ÃĆ    361          Lk                 Arrowhead
## 5 H2030 110196632486      ÃĆ            ÃĆ      ÃĆ            ÃĆ Artesian Lake County Park
## 6 H2030 110434989248      ÃĆ            ÃĆ    361          Lk                  Backbone
##                          NAME ISLOCAL AREALAND AREAWATER     CENTLAT     CENTLON
## 1              Ackerman Cut       N        0    105436 +42.7714854 -91.0771683
## 2                 Allen Crk       N        0     78271 +41.4781465 -95.9224417
## 3              Alligator Lk       N        0     52834 +42.0869879 -90.1785601
## 4              Arrowhead Lk       N        0    114486 +42.2967329 -95.0512242
## 5 Artesian Lake County Park       N        0     99969 +42.1501453 -94.6802219
## 6               Backbone Lk       N        0    455483 +42.6081368 -91.5457812
##      INTPTLAT    INTPTLON
## 1 +42.7714854 -91.0771683
## 2 +41.4702610 -95.9181765
## 3 +42.0869879 -90.1785601
## 4 +42.2967329 -95.0512242
## 5 +42.1501453 -94.6802219
## 6 +42.6017186 -91.5372706

### transform the data to the desired format and pull the information of interest
tigeria$NAME <- as.character(tigeria$NAME)

## pull the des moines river data
dsmriv <- tigeria[tigeria$NAME == "Des Moines Riv", ]

poly.coords <- as.data.frame(dsmriv[, c("INTPTLAT", "INTPTLON")])
row.names(poly.coords) <- NULL
names(poly.coords) <- c("x", "y")

poly.coords[, 1] <- as.numeric(as.vector(poly.coords[, 1]))
poly.coords[, 2] <- as.numeric(as.vector(poly.coords[, 2]))

## overlay the des moines river area over the iowa plot

# qplot(y,x,data=poly.coords.ia)+geom_point(data=poly.coords,aes(x=y,y=x),color='red')

# qmap(location = 'iowa', zoom=7, maptype = 'hybrid') + geom_point(data=poly.coords,
# mapping=aes(x=y, y=x), size=2)

qmap(location = "iowa", zoom = 7, maptype = "hybrid") + geom_point(data = poly.coords.ia, mapping =
    y = x), size = 2) + geom_point(data = poly.coords, mapping = aes(x = y, y = x), size = 2,
    color = "red")
```

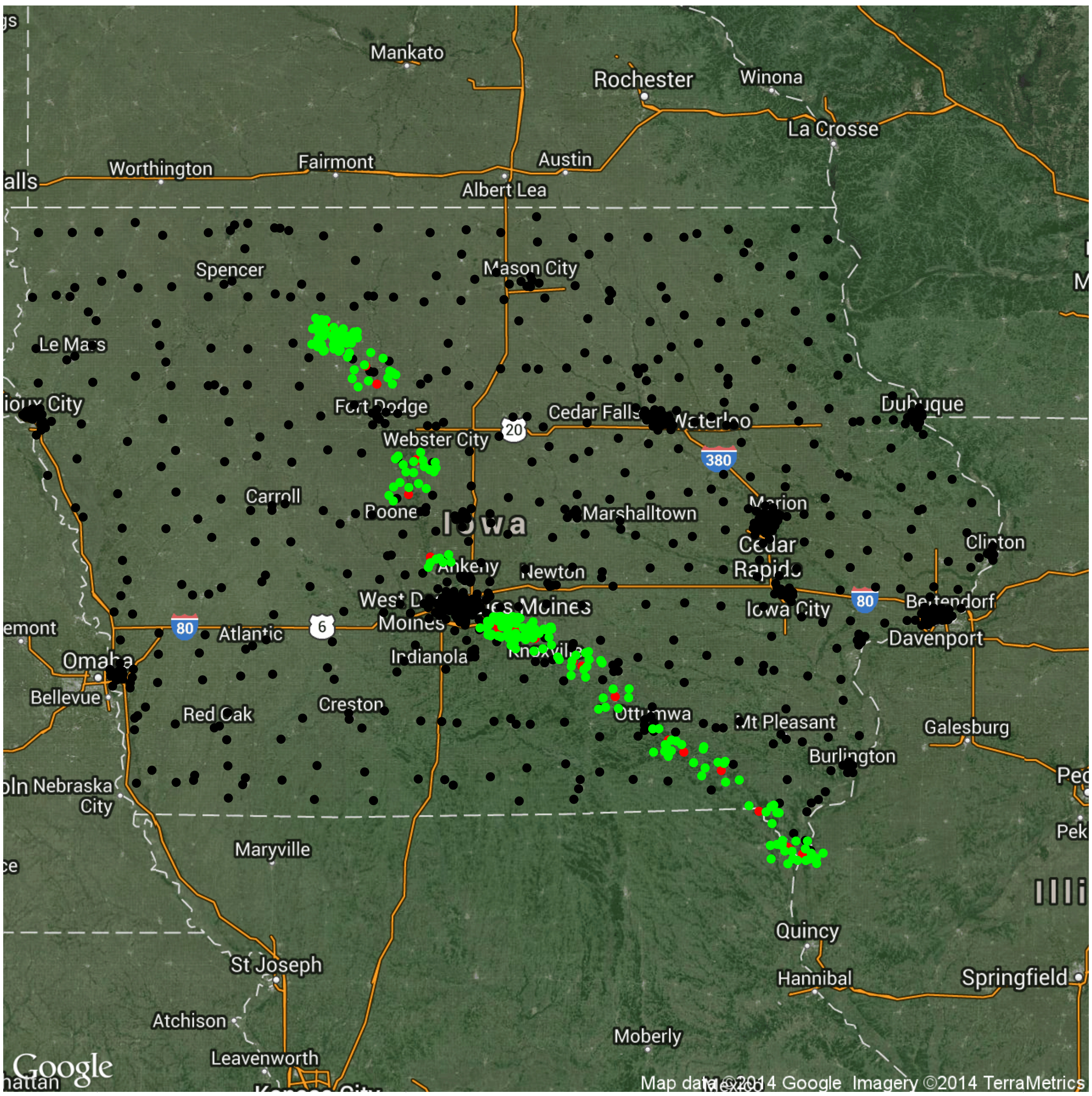


The next challenge is to mimic the CEAP sample. The Des Moines River data contains points only on the river, but not on the entire watershed. We are using the *plyr* package in R to expand this sample of points, in order to capture the region along the river. The approach we decide to use is to add noise to each of the existing points, using the *jitter* function. Also, we create six more points in the nearest neighborhood of the existing points. A map of the final region is presented below.

Note. This approach of constructing a larger sample, including the Des Moines River sample points and some of the surrounding, may be questionable because it is not certain whether the resulting sample of points is a subset of the points in the CDL data. Further approaches are discussed in the *Future work* section in this report.

```
## create a region of interest around the des moines river

set.seed(2013)
add.poly.coords <- as.data.frame(t(ldply(llply(poly.coords, function(x) {
    ldply(x, function(y) jitter(rep(y, 7), 0.075))
```

```
}), unlist)[, -1]))
names(add.poly.coords) <- c("x", "y")

# qplot(y,x,data=poly.coords.ia)+ geom_point(data=poly.coords,aes(x=y,y=x),color='red')+
# geom_point(data=add.poly.coords,aes(x=y,y=x),color='green')

qmap(location = "iowa", zoom = 7, maptype = "hybrid") + geom_point(data = poly.coords.ia, mapping =
    y = x), size = 2) + geom_point(data = poly.coords, mapping = aes(x = y, y = x), size = 2,
    color = "red") + geom_point(data = add.poly.coords, mapping = aes(x = y, y = x), size = 2,
    color = "green")
d <- nrow(add.poly.coords)
d2 <- nrow(poly.coords)
```



Once we have the desired region and a fairly sizeable sample (a number of 224), we extract the information from the CDL data. First, we create the matrix of point coordinates in coordinate reference system (CRS) using the attributes of the raster objects described previously. Next we use the *cell-*

FromXY function to extract the pixel count information available in the CDL data for the points in the region, for each year of interest. The results are numeric vectors of size 224, containing the crop codes for the points. We construct the proportions of crops, by category, by year and the results are in the code chunk below.

First we consider the original sample of only 32 points, representing the Des Moines River.

```
# consider first the Des Moines river only

# get the coordinates in CRS
loc.newcoords <- project(cbind(poly.coords[, 2], poly.coords[, 1]), proj = "+proj=aea +lat_1=29.5 +

# gets the values of the pixels
cdl.pts3 <- cdl.ia03[cellFromXY(cdl.ia03, loc.newcoords)]
table(cdl.pts3)/length(cdl.pts3[-which(is.na(cdl.pts3))]) * 100

## cdl.pts3
##     25     61     63     83    176
##  3.226  3.226  9.677 19.355 64.516

cdl.pts4 <- cdl.ia04[cellFromXY(cdl.ia04, loc.newcoords)]
table(cdl.pts4)/length(cdl.pts4[-which(is.na(cdl.pts4))]) * 100

## cdl.pts4
##      1     70     82     83    176
##  3.226  3.226  3.226 58.065 32.258

cdl.pts5 <- cdl.ia05[cellFromXY(cdl.ia05, loc.newcoords)]
table(cdl.pts5)/length(cdl.pts5[-which(is.na(cdl.pts5))]) * 100

## cdl.pts5
##   5  63  81  82  83  87 176
## Inf Inf Inf Inf Inf Inf Inf

cdl.pts6 <- cdl.ia06[cellFromXY(cdl.ia06, loc.newcoords)]
table(cdl.pts6)/length(cdl.pts6[-which(is.na(cdl.pts6))]) * 100

## cdl.pts6
##   1 111 131 176 190 195
## Inf Inf Inf Inf Inf Inf

cdl.pts7 <- cdl.ia07[cellFromXY(cdl.ia07, loc.newcoords)]
table(cdl.pts7)/length(cdl.pts7[-which(is.na(cdl.pts7))]) * 100

## cdl.pts7
## 111 190 195
## Inf Inf Inf
```

Next, we consider the larger samples, created by adding the surrounding points to the Des Moines River sample points.

```r
# consider next the Des Moines river and the surrounding region

# get the coordinates in CRS
loc.newcoords <- project(cbind(add.poly.coords[, 2], add.poly.coords[, 1]), proj = "+proj=aea +lat_

# gets the values of the pixels
cdl.pts3 <- cdl.ia03[cellFromXY(cdl.ia03, loc.newcoords)]
table(cdl.pts3)/length(cdl.pts3[-which(is.na(cdl.pts3))]) * 100

## cdl.pts3
##       1       5      25      28      36      61      63      82      83     176
## 23.9024 30.7317  1.4634  0.4878  2.4390  6.3415  4.3902  2.4390  3.9024 23.9024

cdl.pts4 <- cdl.ia04[cellFromXY(cdl.ia04, loc.newcoords)]
table(cdl.pts4)/length(cdl.pts4[-which(is.na(cdl.pts4))]) * 100

## cdl.pts4
##       1       5      25      28      36      61      63      70      81      82      83
## 28.7805 22.4390  1.9512  0.4878  2.9268  2.9268 11.2195  0.4878  0.9756  2.9268  4.8780
##      88     176
##  0.4878 19.5122

cdl.pts5 <- cdl.ia05[cellFromXY(cdl.ia05, loc.newcoords)]
table(cdl.pts5)/length(cdl.pts5[-which(is.na(cdl.pts5))]) * 100

## cdl.pts5
##       1       5      36      44      61      63      70      82      83      88     176
## 25.3659 26.3415  1.4634  0.4878  2.9268 13.6585  0.4878  1.9512  4.3902  1.4634 21.4634

cdl.pts6 <- cdl.ia06[cellFromXY(cdl.ia06, loc.newcoords)]
table(cdl.pts6)/length(cdl.pts6[-which(is.na(cdl.pts6))]) * 100

## cdl.pts6
##       1       5      28      36      61     111     121     122     141     143     176
## 27.3171 25.3659  0.4878  2.4390  1.9512  4.3902  3.9024  0.4878 14.6341  0.4878 16.0976
##     190     195
##  1.4634  0.9756

cdl.pts7 <- cdl.ia07[cellFromXY(cdl.ia07, loc.newcoords)]
table(cdl.pts7)/length(cdl.pts7[-which(is.na(cdl.pts7))]) * 100

## cdl.pts7
##       1       5     111     121     141     176     190     195
## 27.8049 18.0488  5.3659  9.7561 12.6829 22.4390  2.9268  0.9756
```

Notice that by considering a larger sample of points, we successfully overlapped with the CDL points sample. That is, we are able to see changes in more than just the crop categories present in the Des Moines River Census sample.

In the following part of this report we present the results in a nicer way, displaying the crop classes corresponding the codes, in order to better understand the changes over this time period. For this, we pull the codes and classes data from the NASS website. These HTML data need more cleaning because there are multiple tables on the page, as well as paragraphs of text. We only need the table containg the codes and the associated classes so we need to filter out some rows and columns from the table pulled using the *readHTMLTable* function. One way of doing this is simply exclude the rows and the columns that are not of interest. Another, more elegant, way of accomplishing this task is to navigate on the source page and use the *XML* root and childern information. However, the source code is not in a compact format, the objects are not uniquely identifiable (there are no options, no values and such). So we can not track the path to the only one table of interest and we use decide to simply exclude some rows and columns. We save the data into a dataframe, containg two columns, one column of codes and one column of classes, and we give the appropriate names.

Next, we merge the codes and classes dataset with the yearly CDL point datasets, saving only the proportion of points in each code. We start by writing a general function to complete this task but, using the *debug/browser* functions in *R* we are able to identify challenges as we continue coding.

```
src <- "http://www.nass.usda.gov/research/Cropland/docs/CDL_2013_crosswalk.htm"
tables <- readHTMLTable(src)
ldply(tables, dim)

##    .id  V1 V2
## 1 NULL 316 13

codes <- tables[[1]]
head(codes)

##   V1
## 1
## 2
## 3
## 4
## 5
## 6
##
## 1                                The 1997-2013 CDLs were recoded and\r\n  re-released in Januar
## 2                        ÃĆÂăcategory named Grass/Pasture (code 176)\r\n  collapses the following
## 3                         ÃĆÂă(code\r\n  171), and Pasture/Hay (code 181). This was done to elimin
## 4                        ÃĆÂăclassified definitionally consistent from\r\n  state to state or year
## 5                        ÃĆÂăthe\r\n  recoding of the entire CDL archive in January 2012 to better
## 6 ÃĆÂăof the category name and code\r\n  changes, please visit the Frequently Asked Questions (F
##   V3 V4   V5   V6   V7   V8   V9  V10  V11  V12  V13
## 1                          <NA> <NA> <NA> <NA> <NA>
## 2                     <NA> <NA> <NA> <NA> <NA> <NA>
## 3                     <NA> <NA> <NA> <NA> <NA> <NA>
## 4                          <NA> <NA> <NA> <NA> <NA>
## 5                          <NA> <NA> <NA> <NA> <NA>
## 6       <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA> <NA>

## keep only the codes/classes information

## one way is to navigate on the source page and use root/children XML ideas but the source
## code is not in a compact format, it is written such that the objects on the page are not
## uniquely identified (no options, values, etc)

url <- "http://www.nass.usda.gov/research/Cropland/docs/CDL_2013_crosswalk.htm"
```

```r
doc <- htmlParse(url)
root <- xmlRoot(doc)

length(xmlChildren(root))

## [1] 2

xmlName(xmlChildren(root)[[2]])

## [1] "body"

length(xmlChildren(xmlChildren(root)[[2]]))

## [1] 21

length(getNodeSet(root, "//table"))

## [1] 1

length(getNodeSet(root, "//@width"))

## [1] 452

length(getNodeSet(root, "//table[@width]"))

## [1] 1

## since we only need the first two columns, we select them using 1:2 as for the rows, we
## need to count the first 9 rows of text and skip them and also we need to through away the
## last chunck of rows

codes <- as.data.frame(codes[-c(1:9, 266:316), 1:2])
names(codes) <- c("code", "class")
codes[, "class"] <- as.character(codes[, "class"])

## function to merge the points and the codes datasets, by the common codes
merging <- function(x) {

    pts <- as.data.frame(x[-which(is.na(x))])
    names(pts) <- "code"
    res <- join(codes, pts, by = "code")
    res2 <- res[match(pts[, "code"], res[, "code"]), ]

    return(table(res2[, "class"])/nrow(pts) * 100)
}

pts37 <- as.data.frame(cbind(cdl.pts3, cdl.pts4, cdl.pts5, cdl.pts6, cdl.pts7))

d <- apply(pts37, 2, merging)  ###or alply

d2 <- laply(d, length)  ###different sizes

allcrops <- unlist(llply(d, function(x) names(x)))
cropmaxno <- length(unique(allcrops))
```

The first challenge is due to the fact that the sizes differ. The data for each year contains information on the crop classes planted that year, a subset of the total number of crop classes.

Total	2003	2004	2005	2006	2007
21	10	13	11	13	8

Table 1: Number of crop classes by year

We write a function using the *join* in *plyr* and the *match* functions in R to complete this task and apply it to the yearly information.

The next challenge is to complete yearly information so that all the years have the same number of crop classes. The approach we decide to take is to introduce missing data for the classes that are missing in a year. Our first option for tools is the *–ply* function from the *plyr* package but we do not find a direct way of completing this task. Also, we try to apply a function that imputes missing values and new crop classes for each year, where the yearly data is an element in a list. This is not successful either because we can not make changes to elements in the list.

```
## bring all the years to same number of crop categoris by imputing NA for the years where a
## crop is missing
dat <- d

laply(dat, function(x) names(x) <- unique(allcrops))  ## doesn't do it automatically


### this function should work!!!!!!!!
llply(dat, function(x) {
    for (i in 1:cropmaxno) if (length(grep(unique(allcrops)[i], names(x))) < 1) {
        x <- c(x, NA)
        names(x)[i] <- unique(allcrops)[i]
    }
})

## the problem is that we cannot assign the changes to each element
```

So we decide to write a function that verifies the matching between the existing classes in a year data and the set of all classes, using regular expressions. In particular, we use the *grep* and *identical* functions. The idea turns out to be the final product idea, however there is one more challenge. Some crop classes have names that include other crop classes, for example *Developed* and *Developed/Low Intensity* or *Developed/Open Space*. The *grep* function verifies for elements in a set, but only as substrings of strings. This function does not verify for a perfect string match. Using a few *if* statements and the *debug/browser* functions we complete the task. The resulting function is applied to the existing data and it creates a list of yearly data, each element containing crop classes and proportion of cultivated land in each class. The data is then combined into a final product dataframe.

Note: Ideally, we would have preferred using the *plyr* package to complete this task, by assigning a function to each element in a list and obtain a dataframe (i.e *dlply* function), however the number of crop classes differs by year (elements in the list have different sizes) and a dataframe can not have columns of different sizes. We would further investigate if there are functions to handle these challenges.

```r
## try
fxn <- function(data) {
    for (x in 1:length(data)) {
        for (i in 1:cropmaxno)
        if (length(grep(unique(allcrops)[i], names(data[[x]]))) < 1) {
            data[[x]] <- c(data[[x]], NA)
            names(data[[x]])[i] <- unique(allcrops)[i]
        }
        paste("change number", data)
    }
    return(data)
}

# fxn(d) -> newd

## it works, except for the last year, where the categories are developed, developed open
## space, developed etc... the grep function fails because developed is subset ...need to
## match exactly...

## next try
fxn <- function(data) {

    # browser()
    for (x in 1:length(data)) {
        for (i in 1:cropmaxno) {

            cnt <- 0
            subset <- grep(unique(allcrops)[i], names(data[[x]]))
            len <- length(subset)

            if (len < 1) {
                data[[x]] <- c(data[[x]], NA)
                names(data[[x]])[length(data[[x]])] <- unique(allcrops)[i]
            }

            if (len >= 1)
                for (j in 1:len) if (!identical(unique(allcrops)[i], names(data[[x]])[[subset[j]]]))
                  cnt <- cnt + 1

            if (cnt >= 1) {
                data[[x]] <- c(data[[x]], NA)
                names(data[[x]])[length(data[[x]])] <- unique(allcrops)[i]
            }
        }

    }
    return(data)
}

newd <- fxn(d)
mode(newd)


## [1] "list"
```

```
newd

## $cdl.pts3
##                   Alfalfa                          Corn                     Developed
##                    2.4390                       23.9024                        2.4390
##    Fallow/Idle\r\n  Cropland                        Forest                 Grass/Pasture
##                    6.3415                        4.3902                       23.9024
##                      Oats      Other Small\r\n  Grains                      Soybeans
##                    0.4878                        1.4634                       30.7317
##                     Water              Christmas Trees                Clouds/No Data
##                    3.9024                            NA                            NA
##           Nonag/Undefined                  Other Crops              Deciduous Forest
##                        NA                            NA                            NA
## Developed/Low\r\n  Intensity    Developed/Open\r\n  Space     Herbaceous\r\n  Wetlands
##                        NA                            NA                            NA
##              Mixed Forest                   Open Water                Woody Wetlands
##                        NA                            NA                            NA
##
## $cdl.pts4
##                   Alfalfa              Christmas Trees                Clouds/No Data
##                    2.9268                        0.4878                        0.9756
##                      Corn                    Developed    Fallow/Idle\r\n  Cropland
##                   28.7805                        2.9268                        2.9268
##                    Forest                Grass/Pasture               Nonag/Undefined
##                   11.2195                       19.5122                        0.4878
##                      Oats      Other Small\r\n  Grains                      Soybeans
##                    0.4878                        1.9512                       22.4390
##                     Water                  Other Crops              Deciduous Forest
##                    4.8780                            NA                            NA
## Developed/Low\r\n  Intensity    Developed/Open\r\n  Space     Herbaceous\r\n  Wetlands
##                        NA                            NA                            NA
##              Mixed Forest                   Open Water                Woody Wetlands
##                        NA                            NA                            NA
##
## $cdl.pts5
##                   Alfalfa              Christmas Trees                          Corn
##                    1.4634                        0.4878                       25.3659
##                 Developed    Fallow/Idle\r\n  Cropland                        Forest
##                    1.9512                        2.9268                       13.6585
##             Grass/Pasture              Nonag/Undefined                   Other Crops
##                   21.4634                        1.4634                        0.4878
##                  Soybeans                        Water                          Oats
##                   26.3415                        4.3902                            NA
##      Other Small\r\n  Grains               Clouds/No Data              Deciduous Forest
##                        NA                            NA                            NA
## Developed/Low\r\n  Intensity    Developed/Open\r\n  Space     Herbaceous\r\n  Wetlands
##                        NA                            NA                            NA
##              Mixed Forest                   Open Water                Woody Wetlands
##                        NA                            NA                            NA
##
## $cdl.pts6
##                   Alfalfa                          Corn              Deciduous Forest
##                    2.4390                       27.3171                       14.6341
## Developed/Low\r\n  Intensity    Developed/Open\r\n  Space    Fallow/Idle\r\n  Cropland
```

```
##                        0.4878                          3.9024                          1.9512
##                 Grass/Pasture     Herbaceous\r\n  Wetlands                    Mixed Forest
##                       16.0976                          0.9756                          0.4878
##                          Oats                      Open Water                        Soybeans
##                        0.4878                          4.3902                         25.3659
##                Woody Wetlands                       Developed                          Forest
##                        1.4634                              NA                              NA
##       Other Small\r\n  Grains                           Water                Christmas Trees
##                            NA                              NA                              NA
##                Clouds/No Data                 Nonag/Undefined                     Other Crops
##                            NA                              NA                              NA
##
## $cdl.pts7
##                          Corn                Deciduous Forest    Developed/Open\r\n  Space
##                       27.8049                         12.6829                          9.7561
##                 Grass/Pasture     Herbaceous\r\n  Wetlands                      Open Water
##                       22.4390                          0.9756                          5.3659
##                      Soybeans                  Woody Wetlands                        Alfalfa
##                       18.0488                          2.9268                              NA
##                     Developed    Fallow/Idle\r\n  Cropland                          Forest
##                            NA                              NA                              NA
##                          Oats      Other Small\r\n  Grains                           Water
##                            NA                              NA                              NA
##               Christmas Trees                  Clouds/No Data                 Nonag/Undefined
##                            NA                              NA                              NA
##                   Other Crops Developed/Low\r\n  Intensity                    Mixed Forest
##                            NA                              NA                              NA
```

```
# debug(fxn(d))

laply(newd, length)  ###works!!!!! all years have same number of crop classes
```

```
## [1] 21 21 21 21 21
```

```
## now combine all the years cdl.pts into one data frame

d <- as.data.frame(newd)
d
```

```
##                                cdl.pts3 cdl.pts4 cdl.pts5 cdl.pts6 cdl.pts7
## Alfalfa                          2.4390   2.9268   1.4634   2.4390  27.8049
## Corn                            23.9024   0.4878   0.4878  27.3171  12.6829
## Developed                        2.4390   0.9756  25.3659  14.6341   9.7561
## Fallow/Idle\r\n  Cropland       6.3415  28.7805   1.9512   0.4878  22.4390
## Forest                           4.3902   2.9268   2.9268   3.9024   0.9756
## Grass/Pasture                   23.9024   2.9268  13.6585   1.9512   5.3659
## Oats                             0.4878  11.2195  21.4634  16.0976  18.0488
## Other Small\r\n  Grains         1.4634  19.5122   1.4634   0.9756   2.9268
## Soybeans                        30.7317   0.4878   0.4878   0.4878       NA
## Water                            3.9024   0.4878  26.3415   0.4878       NA
## Christmas Trees                      NA   1.9512   4.3902   4.3902       NA
## Clouds/No Data                       NA  22.4390       NA  25.3659       NA
## Nonag/Undefined                      NA   4.8780       NA   1.4634       NA
```

```
## Other Crops                                   NA        NA        NA        NA        NA
## Deciduous Forest                              NA        NA        NA        NA        NA
## Developed/Low\r\n  Intensity                  NA        NA        NA        NA        NA
## Developed/Open\r\n  Space                     NA        NA        NA        NA        NA
## Herbaceous\r\n  Wetlands                      NA        NA        NA        NA        NA
## Mixed Forest                                  NA        NA        NA        NA        NA
## Open Water                                    NA        NA        NA        NA        NA
## Woody Wetlands                                NA        NA        NA        NA        NA
```

Notice that if we simply use the *as.data.frame* function to convert a dataframe from the list of yearly crop proportions, the results do not match the correct crop classes. That is, the proportions for year 2003 match, but the the proportions for the following years are simply binded to the first year, columnwise, but without matching the crop classes. This is due to the construction of the list, we added the crop classes that were not observed in a year at the end of that year data, so the order in which the crop classes show up in each year is not the same for all years. Hence, before presenting the results, we need to do more work in cleaning the data.

```
newd2 <- newd
# d <- as.data.frame(llply(newd2,function(x) { y <- as.data.frame(x) y <-
# y[order(row.names(y)),]})) row.names(d) <- sort(row.names(as.data.frame(newd2[[1]])))

d <- ldply(newd2, function(x) {
    y <- as.data.frame(x)
    y <- y[order(row.names(y)), ]
})
d <- t(d[, -1])
row.names(d) <- sort(unique(allcrops))
d <- cbind(row.names(d), round(d, 4))
```

3 Results

We present summaries for all the crop classes, by year in the Table 2.

4 Conclusions/ Future work

In the previous section we described an approach of constructing a larger sample of points surrounding the Des Moiner River in order to increase the chance of overlapping with the CDL sample of points and increase the number of crop classes observed in each year. There are other ways of constructing this sample, in the idea of 'mimicking' the CEAP sample. One could consider sampling points from the Census Iowa sample of points, close to the points in the Des Moines River sample by introducing a distance matrix, per say. Another approach could be to use a sample of the CDL points close to the points in the Des Moines River sample. We did not investigate any of these possibilities any further because the next step in our work would be in utilizing the real CEAP sample of points, that is available in CSSM, but not publicly available to be used for this project.

Hence, the following conclusions hold only for the region we have described in the previous section. Whether they still hold for the real CEAP data needs future investigation.

Crop Class	2003	2004	2005	2006	2007
Alfalfa	2.439	2.9268	1.4634	2.439	NA
Christmas Trees	NA	0.4878	0.4878	NA	NA
Clouds/No Data	NA	0.9756	NA	NA	NA
Corn	23.9024	28.7805	25.3659	27.3171	27.8049
Deciduous Forest	NA	NA	NA	14.6341	12.6829
Developed	2.439	2.9268	1.9512	NA	NA
Developed/Low Intensity	NA	NA	NA	0.4878	NA
Developed/Open Space	NA	NA	NA	3.9024	9.7561
Fallow/Idle Cropland	6.3415	2.9268	2.9268	1.9512	NA
Forest	4.3902	11.2195	13.6585	NA	NA
Grass/Pasture	23.9024	19.5122	21.4634	16.0976	22.439
Herbaceous Wetlands	NA	NA	NA	0.9756	0.9756
Mixed Forest	NA	NA	NA	0.4878	NA
Nonag/Undefined	NA	0.4878	1.4634	NA	NA
Oats	0.4878	0.4878	NA	0.4878	NA
Open Water	NA	NA	NA	4.3902	5.3659
Other Crops	NA	NA	0.4878	NA	NA
Other Small Grains	1.4634	1.9512	NA	NA	NA
Soybeans	30.7317	22.439	26.3415	25.3659	18.0488
Water	3.9024	4.878	4.3902	NA	NA
Woody Wetlands	NA	NA	NA	1.4634	2.9268

Table 2: Proportion of land by crop class, by year.

Notice that 8 out of the 21 total crop classes are missing for all the years, for this particular region. Also, some of the crop classes are present only in some years, for example *Christmas trees* are not recorded, for this region, in 2003, 2006 and in 2007.

There are missing records in 2004 and 2005 due to clouds and to undefined records. These proportions are not significantly larger than most of the crop proportions. The records from 2003, 2006 and 2007 seem to have better records of crop classes, not being affected of missing data.

The proportions of crops, for each class, change over the years. For example, there are lower observed proportions of Corn in 2003 and 2005, then the observed proportions of Corn in 2004, 2006 and 2007. Also, there are higher proportions of Grass/Pasture, Fallow/Idle Cropland and Soybean in 2003, then in the following years. On the other hand, the proportions of Forest and Water increase from 2003 to the following years. Since the CEAP survey frame was constructed in 2003 and the data was collected in the following four years, these changes may lead to frame coverage problems that need to be investigated.

We have successfully worked with big data in different formats (.tif, web, coordinates in different measurement system, text and general expressions use) and developed tools to manipulate and extract useful information out of it. The next step is to utilize the tools constructed in this project to analyze changes in the real CEAP data. We would also like to investigate the CDL data as possible source of covariates in CEAP small area models. Another possible extension of this work with be the implementation of a web application, such as shiny. This would allow the users to better interact with CEAP data from yearly surveys, from different regions across United States.

5 Final note

This work is completed under the theme of reproducible research and we have created a GitHub repository containing all the files. There have been challenges in maintaining a flow in the pull/commit/push steps throughout the work. Not only we have filled up the *U:* drive space when downloading the large CDL data, but also we have confused GitHub when trying to upload the data on the web repository. When we tried to upload the data, the commit went through but we were not able to push. Hence, we had to use linux commands, such as *git commit -m "commit message"* and take control in the command line. Also, the terminal server ts2.stat.iastate.edu crashed due to heavy load one day, when we were completing a commit. The files remained checked when git crushed. Once again, we used the linux commands, such as *rm -f ./.git/index.lock* to unlock everything and be able to commit and push. Finally, in order to remove the extra folder created from the *.pdf* compilation using *knitr*, we used the command line *git rm -r –cached 'foldername'*. For example, *git rm -r –cached cache* for the cached chunks (web scrapping the census data takes some time) and *git rm -r –cached figure* for the figures (making the google maps takes some time, too).

This report is available in my current GitHub repository, `https://github.com/andreeae/STAT585X-Project`, under *FinalReport.Rnw*. The .pdf and the data are available upon request, as it can not be updated due to large size.

6 References

1. Hijmans, R. (2014), "Package 'plyr'," *http://cran.r-project.org/web/packages/plyr/plyr.pdf*
2. Wickham, H. (2014), "Package 'raster'," *http://cran.at.r-project.org/web/packages/raster/raster.pdf*