Accessing data from D-PLACE in R

Introduction

1. Go to the D-Place data repository on github: https://github.com/D-PLACE/dplace-data
2. Open the "datasets" folder.
3. Choose your dataset - we'll look at "EA" (the Ethnographic Atlas) for this example.

You'll see several files. Let's download "data.csv". This contains the main coding data for all societies for all variables. It's in a long table format.

Downloading files is a bit tricky on Github:

4. Click on "data.csv". This will bring you to a page that summarises the file. It's not a link to the raw file itself (sometimes people try to download this link and end up with a web page rather than a proper file).
5. Click the "Download" button to see the raw data, or if you can't see a "Download" there may be a "Raw" button. But probably you want to download this file. Right-click "Download"/"Raw" and select "Save link as ..." (or "Save target as ...", depending on your browser).

Let's look at the data:

```
d = read.csv("data.csv",stringsAsFactors = F)
knitr::kable(head(d[,1:5]))
```

soc_id	sub_case	year	var_id	code
Aa1	Nyai Nyae region	1950	EA001	8
Aa2	with special reference to Central Dorobo	1920	EA001	4
Aa3	Gei/Khauan tribe	1840	EA001	1
Aa4		1920	EA001	4
Aa5	Epulu net-hunters, Ituri forest	1930	EA001	3
Aa6		1920	EA001	1

This is a long-format file: Each row is a single observation of a single ethnographic variable for a single society.

- The `soc_id` variable shows the HRAF code for the society.
- The `var_id` shows shows an id for the ethnographic variable being coded (e.g. "EA001" is Subsistence economy: gathering).
- The `code` variable shows the code - an number that relates to the actual value for the ethnographic variable for that society (e.g. for "EA001", "8" means "76 to 85 percent dependence on gathering").

There are also columns for comments and bibliographic information.

Let's say we're only interested in the ethnographic variable "Transfer of residence at marriage: prevailing pattern" (which I happen to know is variable number "EA011"). We can sub-set the data to get just that:

```
marriageLocality = d[d$var_id=="EA011",]
```

Linking data

The data above is what we want, but it's hard to interpret. The Github repository has some other files that will help us link the data

- societies.csv: Useful information for how to map between society IDs (HRAF, D-PLACE, Glottolog), and latitude/longitude.
- variables.csv: A mapping between ethnographic variable IDs (e.g. "EA011") and names of the variable (e.g. "Transfer of residence at marriage: prevailing pattern").
- codes.csv: A list of the meaning of each code.

Let's load that data:

```
codes = read.csv("codes.csv",stringsAsFactors = F)
knitr::kable(head(codes))
```

var_id	code	description	name
EA001	0	Zero to 5 percent dependence	0-5%
EA001	1	6 to 15 percent dependence	6-15%
EA001	2	16 to 25 percent dependence	16-25%
EA001	3	26 to 35 percent dependence	26-35%
EA001	4	36 to 45 percent dependence	36-45%
EA001	5	46 to 55 percent dependence	46-55%

Each line is an interpretation of a single code. Let's see the meanings of the codes for marriage locality:

```
knitr::kable(codes[codes$var_id=="EA011",],)
```

	var_id	code	description	name
106	EA011	NA	Missing data	Missing data
107	EA011	1	Wife to husband's group (patrilocal, virilocal) or wife to husband's mother's brother's household (avunculocal)	Wife to husband
108	EA011	2	Couple to either group or neolocal	Ambi/neo-local
109	EA011	3	Husband to wife's group	Husband to wife
110	EA011	9	Nonestablishment of a common household, i.e., where both spouses remain in their natal households, sometimes called "duolocal" or "natolocal" residence	Separate

We can add the "name" of each code to our main data set, in a new variable called `localityType` (this is arbitrary, we could call it `marriageLocality` or something).

```
# Get only codes for marriage locality
mcCodes = codes[codes$var_id=="EA011",]
# Match to names
marriageLocality$localityType =
  mcCodes[match(marriageLocality$code, mcCodes$code),]$name
```

We now have meaningful labels for the codes:

```
head(marriageLocality[,c("soc_id","code","localityType")])
```

```
##      soc_id code    localityType
## 2583    Aa1    2  Ambi/neo-local
```

```
## 2584    Aa2    1 Wife to husband
## 2585    Aa3    1 Wife to husband
## 2586    Aa4    2  Ambi/neo-local
## 2587    Aa5    2  Ambi/neo-local
## 2588    Aa6    1 Wife to husband
```

Let's summarise the distribution of types:

```
table(marriageLocality$localityType)
```

```
##
##  Ambi/neo-local Husband to wife    Missing data        Separate Wife to husband
##             294             103              24                8             862
```

Note that "Missing data" is a type, and it's not always coded as "NA".

Other data

We can continue matching data to add other elements. Let's download the "societies.csv" file, load it, and use it to add data on the society name and location, using the society id:

```
soc = read.csv("societies.csv",stringsAsFactors = F)
marriageLocality$SocName =
  soc[match(marriageLocality$soc_id,
            soc$id),]$pref_name_for_society

marriageLocality$Lat =
  soc[match(marriageLocality$soc_id,
            soc$id),]$Lat
marriageLocality$Long =
  soc[match(marriageLocality$soc_id,
            soc$id),]$Long
marriageLocality$glottocode =
  soc[match(marriageLocality$soc_id,
            soc$id),]$glottocode
```

We can now see data that's more human-readable!

```
head(marriageLocality[,c("soc_id","SocName","code","localityType","Lat","Long")])
```

```
##      soc_id  SocName code    localityType Lat Long
## 2583    Aa1    !Kung    2  Ambi/neo-local -20   21
## 2584    Aa2   Dorobo    1 Wife to husband   0   36
## 2585    Aa3     Nama    1 Wife to husband -26   18
## 2586    Aa4 Bergdama    2  Ambi/neo-local -22   16
## 2587    Aa5    Mbuti    2  Ambi/neo-local   2   28
## 2588    Aa6  Sandawe    1 Wife to husband  -5   36
```

Combining datasets

We can continue this process to extract other variables and combine them. For example, if we wanted to compare marriage locality to the type of transactions at marriage.

Here's a template for doing that:

```
# Match code names
marriageTransactions = d[d$var_id=="EA006",]
mtCodes = codes[codes$var_id=="EA006",]
```

```
# Match to names to create a new variable
#  "transactionType"
marriageTransactions$transactionType =
  mtCodes[match(marriageTransactions$code, mtCodes$code),]$name
```

Now we can combine the two datasets into one.

```
# Make a new dataset that's a copy of marriageLocality
m = marriageLocality
# Combine with marriage
m$transactionType =
  marriageTransactions[match(
    m$soc_id,
    marriageTransactions$soc_id),]$transactionType
```

We now have a single dataset `m` with two variables:

```
knitr::kable(head(m[,c("soc_id","SocName","localityType","transactionType")]))
```

	soc_id	SocName	localityType	transactionType
2583	Aa1	!Kung	Ambi/neo-local	Bride-service
2584	Aa2	Dorobo	Wife to husband	Bride-wealth
2585	Aa3	Nama	Wife to husband	Gift exchange
2586	Aa4	Bergdama	Ambi/neo-local	Bride-service
2587	Aa5	Mbuti	Ambi/neo-local	Woman exchange
2588	Aa6	Sandawe	Wife to husband	Bride-wealth

And we can compare the frequencies:

```
knitr::kable(table(m$localityType, m$transactionType))
```

	Bride-service	Bride-wealth	Dowry	Gift exchange	Insignificant	Missing data	Token bride-wealth	Woman exchange
Ambi/neo-local	47	59	14	25	132	3	12	2
Husband to wife	23	9	1	1	57	2	9	1
Missing data	0	11	0	0	6	7	0	0
Separate	0	4	0	0	3	0	0	1
Wife to husband	55	574	28	39	77	7	47	35

Note that there is some missing data - not all societies are coded for both variables. We might want to remove these, e.g.:

```
m = m[m$localityType!="Missing data",]
m = m[m$transactionType!="Missing data",]
```

Here's a template for completeness:

```
# Load data
d = read.csv("data.csv",stringsAsFactors = F)
codes = read.csv("codes.csv",stringsAsFactors = F)

# get Marriage Locality
marriageLocality = d[d$var_id=="EA011",]
# Get only codes for marriage locality
mcCodes = codes[codes$var_id=="EA011",]
# Match to names to create a new variable
#  "localityType"
marriageLocality$localityType =
  mcCodes[match(marriageLocality$code, mcCodes$code),]$name

# Get Marriage transactions
marriageTransactions = d[d$var_id=="EA006",]
mtCodes = codes[codes$var_id=="EA006",]
# Match to names to create a new variable
#  "transactionType"
marriageTransactions$transactionType =
  mtCodes[match(marriageTransactions$code, mtCodes$code),]$name

# Combine into one data frame
m = marriageLocality
# Combine with marriage
m$transactionType =
  marriageTransactions[match(
    m$soc_id,
    marriageTransactions$soc_id),]$transactionType
```