**A serious, decades-long attempt to understand different peoples’ values**

David Hood (@Thoughfulnz) has been posting some interesting snippets of analysis using the World Values Survey data (like [this example](https://twitter.com/Thoughtfulnz/status/1088339318752464897)). This inspired me to have a look at the data myself; something that’s been on my to-do list for years. I have analysed it before, but a long while ago and for only a single specific purpose, and I wanted to get a more general overview of it.

The [World Values Survey](http://www.worldvaluessurvey.org/WVSContents.jsp) is an amazing multi-decade (started in 1981) network of scientists doing their best to create standard, internationally comparable, nationally representative data on values and their impact on social and political life.

It’s an impressive resource, with six waves of data already published and a seventh in preparation. I’ve only had a few hours today to look at it and that’s enough to make me want to come back. But let’s start with my finishing point for today – a big graphic of around 40 of the questions from the various surveys, showing how much people in different countries agree with various statements:

It really needs a big screen of course. Click on the image to open it in its own tab of the browser.

I find this fascinating and hope you do to. It’s a graphic that repays close attention, and the more I look at it the more interesting snippets I find. The ordering of the graphic with the most “agreed-with” statements at the top and the most “agreeing” countries on the right is crucial. It means that you can look up or across the graphic and look the squares of unusual colour that indicate exceptions. Here’s just some of the things that I noticed:

* The United States stands out for its residents agreement with “Under some conditions, war is necessary to obtain justice”
* The people of Hong Kong don’t agree that they are autonomous individuals
* The people of Sweden are inclined to agree that all religions should be taught in public schools; and to disagree that “when jobs are scarce, employers should give priority to people of this country:
* Residents of former Soviet Union countries (Belarus, Russia, Kazakhstan, Ukraine) stand out for not agreeing with “I see myself as part of my local community”

**Data wrangling**

I have in mind the plan to eventually grab all six waves of the World Values Survey, as the real benefit of this sort of data comes not just from cross-country comparisons but comparisons over time. I also wanted a way of wrangling the data that would lend itself to efficient analysis of multiple questions at once. Both of these needs pushed me away from the common way of treating survey with one row per respondent, one column per question to a more efficient data model as used in data warehousing.

At first I did this all in R, but then I found that when I got to the analysis stage and wanted to join my 38 million row fact table to the 90,000 row dimension table I ran out of memory. I needed a database so that hard disk could be used when the going gets tough, and to draw on all the optimisation built into that software for exactly this job. So I ended up with this toolchain (which is a very common one for me):

* data import, cleaning and some normalising in R
* upload into a database with the right disciplines and constraints, and (sometimes, not today) finishing off the cleaning and normalising
* for analysis, do the heavy lifting in SQL queries but bring filtered or aggregated data into R for any statistical modelling or for presentation.

Here’s the R code for the first step. It draws first on code by David Hood for extracting the infromation on questions and answers that was encoded in the original file as attributes; then moves on to split the data into four tables representing respondents, questions, possible responses and (finally) the 38 million combinations of respondent – question – response combinations in a single long, thin table.

library(tidyverse)

library(frs)

library(data.table)

library(odbc)

library(viridis)

library(ggthemes)

library(scales)

# correct citation must be used WV6\_Data\_R\_v20180912; .

# Obtain data from here:

# http://www.worldvaluessurvey.org/WVSDocumentationWV6.jsp

# including filling in a form re appropriate use etc.

# load in data:

wvs <- readRDS("../data/F00007762-WV6\_Data\_R\_v20180912.Rds")

#--------------extract labels and questions-----------

# label code from David Hood @Thoughtfulnz https://gist.github.com/thoughtfulbloke/8d6e016a74039c030ba3871dca36c19c

unlabel <- function(x){

codes <- attr(x, "labels")

if(is.null(codes)) {

print(paste("No labels with",attr(x, "label")))

return(x)

}

df <- data.frame(y = as.numeric(x), stringsAsFactors = FALSE)

replacement <- data.frame(

y = as.numeric(codes),

newform = names(codes),

stringsAsFactors = FALSE

)

df2 <- df %>% left\_join(replacement, by="y") %>%

mutate(newform = ifelse(is.na(newform), as.character(x), newform))

if(length(x) == length(df2$newform) &

sum(is.na(x)) == sum(is.na(df2$newform))){

return(df2$newform)

} else {

print(paste("Problem with",attr(x, "label")))

return(x)

}

}

wv <- wvs %>%

mutate\_all(unlabel)

labs <- wvs %>%

map\_chr(~attributes(.)$label) %>%

make.names(unique = TRUE)

names(wv) <- labs

# End of David Hood's tidying code

#=============turn into a star schema===================

# we need to make our own ID because nothing in the data seems to be a unique identifier

wv$respondent\_id <- 1:nrow(wv)

# Define the variables that will eventually be in the respondent dimension table.

# This is things about the respondent and the interview and wave, rather than questions

# they've answered.

respondent\_vars <- c("Wave", "Country.Code", "Country.regions..with.split.ups.",

"Interview.number", "Study", "Unified.respondent.number",

"X1000.equilibrated.weight", "X1500.equilibrated.weight",

"Country...wave...study...set...year", "Nation.Wave", "Nation.Year",

"COW.Country.Code", "Weight.for.overal.secular.values",

"Weight.for.Emancipative.values", "Weight", "Weight.with.split.ups",

"Questionnaire.version", "Date.of.interview", "Survey.year", "respondent\_id")

#--------long thin version that will be the basis of the main fact table----------

# we want all the original columns (ie questions), \*except\* those

# that are going to end up in the respondent dimension table. But

# we need respondent\_id as that is going to link the two.

wvt <- wv[ , c(names(wv)[!names(wv) %in% respondent\_vars] , "respondent\_id")] %>%

gather(question, response, -respondent\_id) %>%

mutate(question = as.factor(question),

response = as.factor(response),

question\_id = as.integer(question),

response\_id = as.integer(response)) %>%

# one mystery NA to Highest.educational.level.attained

filter(!is.na(response))

# note that this version is 38 million rows long

# extract the questions and do a bit of cleaning:

d\_questions <- wvt %>%

distinct(question\_id, question) %>%

mutate(question = gsub("..", ": ", question, fixed = TRUE),

question = gsub(".", " ", question, fixed = TRUE),

question = str\_squish(question),

question = gsub("country s ", "country's ", question, fixed = TRUE),

question = gsub("Mother s ", "Mother's ", question, fixed = TRUE),

question = gsub("Father s ", "Father's ", question, fixed = TRUE)) %>%

separate(question, into = c("super\_broad", "very\_broad", "broad\_question", "mid\_question", "narrow\_question"),

fill = "left", remove = FALSE, sep = ":") %>%

mutate(narrow\_question = str\_squish(narrow\_question),

mid\_question = str\_squish(mid\_question),

broad\_question = str\_squish(broad\_question))

# extract the responses and do some cleaning and add some

# classifications that might be useful:

d\_responses <- wvt %>%

distinct(response\_id, response) %>%

# cleaning

mutate(response = str\_squish(gsub("Dont ", "Don't ", response))) %>%

# creating alternative versions of some of the responses:

mutate(response\_no\_commas = gsub(",", "", response, fixed = TRUE),

response\_no\_spaces = gsub(" ", "", response\_no\_commas, fixed = TRUE),

response\_lower\_case = tolower(response),

response\_numeric = suppressWarnings(as.numeric(response\_no\_spaces)),

response\_any\_numerals = ifelse(is.na(response\_numeric),

str\_extract(response\_no\_spaces, "[0-9]+"),

response\_numeric),

response\_class = ifelse(is.na(response\_numeric), "character", "numeric"),

agrees = as.numeric(response\_lower\_case %in%

c("strongly agree", "agree", "completely agree", "agree strongly")),

important = as.numeric(response\_lower\_case %in%

c("very important", "important", "absolutely important", "rather important")),

trust = as.numeric(response\_lower\_case %in%

c("trust somewhat", "trust completely", "most people can be trusted")),

often = as.numeric(response\_lower\_case %in%

c("fairly often", "very often", "often", "quite frequently", "very frequently", "frequently")),

like\_me = as.numeric(response\_lower\_case %in%

c("very much like me", "like me", "somewhat like me", "a little like")),

interested = as.numeric(response\_lower\_case %in%

c("somewhat interested", "very interested", "respondent was somewhat interested",

"respondent was very interested")),

satisfied = as.numeric(response\_lower\_case %in%

c("completely satisfied", "fairly satisfied", "very satisfied", "strongly satisfied")),

happy = as.numeric(response\_lower\_case %in%

c("rather happy", "very happy")),

respect = as.numeric(response\_lower\_case %in%

c("fairly much respect")),

justifiable = as.numeric(response\_lower\_case %in%

c("Always justifiable")),

invalid = as.numeric(response\_lower\_case %in%

c("no answer") |

grepl("not asked", response\_lower\_case) |

grepl("don't know", response\_lower\_case) |

grepl("unsure", response\_lower\_case) |

grepl("unknown", response\_lower\_case) |

grepl("inapplicable", response\_lower\_case) |

grepl("dropped out survey", response\_lower\_case) |

grepl("inappropriate response", response\_lower\_case) |

grepl("missing", response\_lower\_case) |

grepl("not applicable", response\_lower\_case))

) %>%

select(response\_id, everything())

# extract all the individual responses

d\_respondents <- wv[ , respondent\_vars] %>%

rename\_all(tolower) %>%

rename\_all(function(x){gsub("\\.+", "\_", x)}) %>%

select(respondent\_id, everything()) %>%

mutate(weight = ifelse(weight == "No weighting", 1, as.numeric(weight)))

# final version of the factless fact table, with no information other than the combinations

# of respondent, question and response:

f\_wvs <- wvt %>%

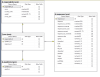
select(respondent\_id, question\_id, response\_id)

One of the key things I’ve done here is coded some of the responses with new classification columns such as agree – which is 1 for every response that is some kind of partly or fully agree, and 0 otherwise. This will let us analyse the multiple questions that use variants of this response set efficiently down the track. The invalid column for every variant on “not asked”, “dropped out”, “don’t know”, is also important.

**Creating a database**

**Defining a data model**

The data model I’ve split the data out into is basically this:



This is the classic data warehousing approach to this sort of data. A long, thin fact table; and shorter, wider dimension tables that hold all the text information and slowly changing information about the entities such as people, questions, etc. For those interested in these things, f\_wvs is a good example of a “factless fact table” – meaning that other than indicating unique combinations of the three dimensions (*this respondent* gave *this response* to *this question*) it has no additional numeric information or “facts”.

There are many many advantages of this data model; as well as being convenient for analysis down the track, one of the advantages is that we can do cleaning and enhancements of the dimension data in a compact way. In this case, I can add classifications to the responses in the 28,000 row d\_responses table much more efficiently (with each unique response represented only once) than if I’d tried to do it to the original data, where each response is repeated each time someone in the original survey ticked it. And those columns of 1s and 0s with names like agree and invalid are going to become crucial for efficient analysis down the track.

Here’s the SQL that I ran in SQL Server to set up that simple empty wvs schema ready to accept the data from R:

USE survey\_microdata

CREATE SCHEMA wvs;

DROP TABLE IF EXISTS wvs.f\_wvs

DROP TABLE IF EXISTS wvs.d\_respondents

DROP TABLE IF EXISTS wvs.d\_responses

DROP TABLE IF EXISTS wvs.d\_questions

CREATE TABLE wvs.d\_respondents(

respondent\_id INT NOT NULL PRIMARY KEY,

wave NVARCHAR(63) NOT NULL,

country\_code NVARCHAR(63) NOT NULL,

weight NUMERIC(20,17) NOT NULL,

survey\_year SMALLINT

)

CREATE TABLE wvs.d\_questions(

question\_id INT NOT NULL PRIMARY KEY,

question NVARCHAR(255) NOT NULL

)

-- response (second column below) isn't unique because of upper / lower case issues for agree strongly

CREATE TABLE wvs.d\_responses(

response\_id INT NOT NULL PRIMARY KEY,

response NVARCHAR(255) NOT NULL,

response\_no\_commas NVARCHAR(255) NOT NULL,

response\_no\_spaces NVARCHAR(255) NOT NULL,

response\_lower\_case NVARCHAR(255) NOT NULL,

response\_numeric NUMERIC(25, 17),

response\_any\_numerals VARCHAR(63),

response\_class VARCHAR(63),

agrees NUMERIC(2,1) NOT NULL,

important NUMERIC(2,1) NOT NULL,

trust NUMERIC(2,1) NOT NULL,

often NUMERIC(2,1) NOT NULL,

like\_me NUMERIC(2,1) NOT NULL,

interested NUMERIC(2,1) NOT NULL,

satisfied NUMERIC(2,1) NOT NULL,

happy NUMERIC(2,1) NOT NULL,

respect NUMERIC(2,1) NOT NULL,

justifiable NUMERIC(2,1) NOT NULL,

invalid NUMERIC(2,1) NOT NULL

)

CREATE TABLE wvs.f\_wvs (

respondent\_id INT NOT NULL,

question\_id INT NOT NULL,

response\_id INT NOT NULL,

FOREIGN KEY (respondent\_id) REFERENCES wvs.d\_respondents(respondent\_id),

FOREIGN KEY (question\_id) REFERENCES wvs.d\_questions(question\_id),

FOREIGN KEY (response\_id) REFERENCES wvs.d\_responses(response\_id)

)

ALTER TABLE wvs.f\_wvs ADD PRIMARY KEY (respondent\_id, question\_id);

**Uploading data from R**

For larger datasets, my preferred approach to getting the data from R into a SQL Server database is:

* define the database tables in SQL (as done above)
* from R, save the tables as pipe-delimited text files with no columns. The fwrite function in data.table package is super-fast at doing this – less than a second for the 38 million rows in f\_wvs in this case.
* upload the data from the local disk to the database server via SQL Server’s bcp bulk import and export tool. This is a common enough task for me that I’ve made a convenience wrapper function bcp() in my [frs R package](https://github.com/ellisp/frs-r-package) of miscellaneous bits and pieces.

Here’s how that upload process looks in R:

#-----------upload to database--------------

fwrite(d\_respondents[ , c("respondent\_id", "wave", "country\_code", "weight", "survey\_year")],

"d\_respondents.txt", sep = "|", col.names = FALSE)

bcp("localhost", "survey\_microdata", "wvs", table = "d\_respondents", file = "d\_respondents.txt")

fwrite(d\_questions[ , c("question\_id", "question")],

"d\_questions.txt", sep = "|", col.names = FALSE)

bcp("localhost", "survey\_microdata", "wvs", table = "d\_questions", file = "d\_questions.txt")

fwrite(d\_responses, "d\_responses.txt", sep = "|", col.names = FALSE)

bcp("localhost", "survey\_microdata", "wvs", table = "d\_responses", file = "d\_responses.txt")

# this is astonishly fast - less than a second to write 38 million rows to disk

fwrite(f\_wvs, "f\_wvs.txt", sep = "|", col.names = FALSE)

# this is more like a 15+ minutes. As well as writing to the database server disk,

# it's checking for uniqueness of primary keys and the foreign key constraints.

bcp("localhost", "survey\_microdata", "wvs", table = "f\_wvs", file = "f\_wvs.txt")

**Analysis**

After all this, the analysis is the easy and fun part. Because this post is already long enough I’m going to just show the creation of that graphic I started with. My workflow at this point is writing SQL queries in SQL Server Management Studio until I’m satisfied I’ve got the right filtering and aggregation; then I copy and paste that query into my R script so everything is in one place. If the SQL gets much more complicated than the below I’d want to save separate SQL scripts rather than copy them into R.

So here it is, code to pull down the appropriately weighted responses for all questions in all countries that have an “agree” part of the response. The code below is specific to my setup in that I have an ODBC data source name “sqlserver” which I use to connect to the localhost database; that was just set up in the Windows ODBC administrator by following the prompts. If you’ve got a remote server it gets a little more complicated.

sql <- "

WITH agree\_questions AS

(SELECT DISTINCT question, c.question\_id

FROM wvs.d\_responses AS a

INNER JOIN wvs.f\_wvs AS b

ON a.response\_id = b.response\_id

INNER JOIN wvs.d\_questions AS c

ON b.question\_id = c.question\_id

WHERE agrees = 1)

SELECT

sum(g.weight \* f.agrees) / sum(g.weight) AS weighted\_agree,

question,

country\_code

FROM wvs.f\_wvs AS d

INNER JOIN agree\_questions AS e

ON d.question\_id = e.question\_id

INNER JOIN wvs.d\_responses AS f

ON d.response\_id = f.response\_id

INNER JOIN wvs.d\_respondents AS g

ON d.respondent\_id = g.respondent\_id

WHERE f.invalid != 1

GROUP BY question, country\_code

ORDER by weighted\_agree DESC"

con <- dbConnect(odbc(), "sqlserver", database = "survey\_microdata")

d <- dbGetQuery(con, sql)

d2 <- d %>%

as\_tibble() %>%

mutate(question = fct\_reorder(question, weighted\_agree)) %>%

mutate(country\_code = fct\_reorder(country\_code, weighted\_agree))

yup <- length(unique(d2$question))

d2 %>%

ggplot(aes(x = country\_code, y = question)) +

theme\_tufte(base\_family = "main\_font") +

geom\_tile(aes(fill = weighted\_agree)) +

scale\_fill\_viridis("Weighted percentage of valid answers that agreed or strongly agreed:", label = percent) +

theme(axis.text.x = element\_text(angle = 45, hjust = 1, size = 8),

legend.position = "bottom",

axis.text.y = element\_text(size = 7)) +

labs(x = "The countries that 'agree' with more statements are on the right.",

y = "The statements most agreed with (when asked) are at the top.\n",

caption = "Source: http://www.worldvaluessurvey.org/WVSDocumentationWV6.jsp, WV6\_Data\_R\_v20180912. Analysis by http://freerangestats.info") +

# draw the countries along the top as a sort of repeat of the x axis:

geom\_text(data = distinct(d2, country\_code), aes(label = country\_code),

y = yup + 1, angle = 45, hjust = 0, size = 2.7) +

expand\_limits(y = yup + 9, x = length(unique(d2$country\_code)) + 3) +

ggtitle("What do people of the world agree with?",

"All questions in the World Values survey with an agree/disagree style of response.")

I’ve ignored statistical issues so far, in particular dealing with sampling uncertainty and the complex survey nature of the data. But I’m all set up to do this when I need to (and I need to understand better the sampling design to do it properly), down the track at some point I hope.

That’s all for now. But I’ll definitely be coming back to this data!