This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 676060

29/9/2017

LONGPOP research output 7.1.

COR technical note for the construction of intermediate data structure (IDS)

Sam Jenkinson, Hideko Matsuo & Koen Matthijs (KU Leuven)

This note concerns the work deliverable of 7.1. regarding concepts and techniques for individual record linkage providing algorithms and relevant syntaxes to construct intergenerational life course trajectories from original or IDS-converted multi-level and multi-source data bases. The ultimate goal of this exercise is to produce new COR-IDS, preparing inputs for the construction of intermediate data structure (IDS), serving for the research output of 7.1. The construction of COR-IDS takes full account of the objective of the LONGPOP project aiming, to create long-term, longitudinal, multi-source and intergenerational data sets. The original input of the COR multi-level, multi-source data file we use here is based on the 2010 version of the COR sample (Matthijs and Moreels 2010). We envisage at least three generations of families (i.e. child, parents and grandparents) will be present in this COR-IDS data and will be available for the wider public use of COR-IDS.

This note consists of the following information based on the process documentation since March 2017.

1. Standardization of core variables for the input of individual and context attribute tables;

2. Standardization and evaluation of individual linkages COR-2010;

3. Specification of standardized variables in meta data format;

4. Individual attributes on core variables from output 1 (indiv data file); and

5. State of art on the population register and how to proceed with indiv-indiv data

Funding Note: This publication reflects only the author's view and the Agency is not responsible for any use that may be made of the information it contains

This document describes the above process outputs in the aforementioned order.

1. Standardization of core variables for the input of individual and context attribute table

In order to ensure the accuracy of the COR-2010 inputs, we have performed an inconsistency check across different sources using the unique identifier of the COR-2010 sample.. We acknowledge that this will be subject to the evaluation of the linkages across different sources, taking into account the newly applied individual identifier (see on 2. standardization and evaluation of individual linkage). We also consider that identifying inconsistencies contributes to the evaluation of linkage too.

Our initial control on core variables (names, gender, event dates, date of birth/death and occupation) based on COR-2010 show the following intra-inconsistencies (see note on 10/3/2017, section 5.1).

When clear differences of information are found, following Mandemakers and Dillon (2010), we give priorities on the sources depending on the types of event and dates we are interested in. For instance, when multiple sources exist for birth dates, preference will be given to birth certificate rather than any other sources such as marriage certificate or death certificate.

2.Standardization and evaluation of individual linkages (COR-2010)

The purpose of this exercise is to examine the quality of the linkage (IDNR) of individual records and attributes across different sources in the COR database. Currently the database is stored as one fully merged database, and not as separate source tables, in line with the best practice database methods, as described by Mandemakers and Dillon (2004). Here they outline how the best practice in historical database management is to retain separate original source tables, in a format as close as possible to a digitised version of the original source. We seek to bring COR closer into line with these guidelines, as best as possible in light of being a pre-existing database with no original source tables or digitised records of the original source, and to examine the accuracy of previous linkages, as well as replicating and improving the original linkage process, in order to prepare for implementing COR IDS.

In order to assess the linkage quality in COR we began by splitting the database into observations from the original historical sources: birth, death, marriage certificate and event (i.e. population register) databases. We then compared the accuracy of vital information of gender, dates and locations of birth, death and names across IDNR (unique identifiers from

previous linkage). This will give us information concerning the size of any errors in linkage currently within the database.

The second step was to relink the database from the original source tables, ignoring the previous identification numbers given during the prior record linkage process. This allowed us to evaluate the process of the original linkage and provide the linkage algorithm for others to use in line with deliverable 7.1. Here initially we use the methods as in line with the initial linkage 2010 (Van Balen). We use a stochastic record linkage method provided by Sariyar and Borg (Sariyar & Borg, 2010) as part of their R package "record linkage" (2016).

The record linkage process provided by the package uses the Fellegi-Sunter Model (Sariyar & Borg, 2010), as seen below

$$w_{\tilde{\gamma}} = \log\left(\frac{P(\gamma = \tilde{\gamma} \mid Z = 1)}{P(\gamma = \tilde{\gamma} \mid Z = 0)}\right).$$

Where the linkage relies on the assumption of conditional probabilities regarding comparison patterns. This works on the probability that a random vector $\gamma = (\gamma 1, \ldots, \gamma n)$, having the value $\tilde{\gamma} = (\tilde{\gamma} 1, \ldots, \tilde{\gamma} n)$, is conditional on the matching status of Z. Where Z=0 stands for a non-match and Z=1 equals match. In the full Fellegi-Sunter model these are used to compute weights which are used in order to discern matches and non-matches. The weights within the package are computed using an expectation maximum (EM) algorithm in line with Haber (1984) and Contiero et al (2005).

We then use common variables across observations sources to calculate the likelihood of a record being a match using a string comparison tool. The selected variables here included given and family names, birth location, birth day, birth month and year separately. These are then used to calculate similarities across different records and to create pairs.

The string comparison used here is the Jarrow-Winkler distance string comparison tool (Winkler, W.E 1990). This function works by measuring the edit distance between two strings and calculates the minimum numbers of single character transpositions to transform one word into another.

The original Jaro distance calculation can be seen below;

$$\text{Jaro dist} = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{|m|}\right)$$

Where m= the number of common characters which are within half of the length of the longer string and t the number of transpositions. The Jaro-Winkler distance is the same, but with the following changes

$$JaroWinkler\ dist = Jari\ dist + l(1 - Jarodist)$$

Where ι represents the common prefix at the start of a string up to a maximum of 4 characters, a standard approach for these purposes.

In addition to this we begin the process using the procedure of blocking (Sariyar & Borg, 2010) to ensure the strictest criteria for record matches, before sequentially relaxing the criteria in order to report the frequency of matches given at a particular matching stringency. Blocking limits the number of matches by ensuring that one particular column or variable is a 100% match. This uses a phonetic tool contained within the package before moving on to assess the individual data contained in other columns using the string comparison tool. This is useful to provide perfect matches but also used to reduce computation time. We begin with one column which is a unique column based on all data contained in the other columns used for linkage. This column contains a string made of all names, birth dates and birth locations. This will then give us the number of matches which are 100% identical across all comparison fields for birth and death certificates. We then slowly remove the number of variables, over three rounds initially here, included in the initial blocking identifier column and report the number of matches and therefore their quality.

The steps are implemented as follows. The first name was split into 5 possible first name variables owing to the likelihood for some to have multiple given names. Initially all 5 of these first names, as well as family name, Birth location in NIS code, birthday, birth month and birth year are included in the blocking criteria. In the second round only the first three given names, surnames, birth places and birth dates are included and in the final round the blocking criteria included only the full birth dates (day, month, year) and two first names and family names.

During this linkage procedure, we created a cleaning function in R. This involved standardising all variables, just for the purposes of linkage, to lower case and removing any white space before or after characters which may have been mistakenly entered by initial data entry. This caused multiple problems on initial linkages as many observations had stray white space before, after and within names making comparisons difficult. In addition erroneous capitalisation was frequently also a problem.

The exercise was carried out first with both birth and death certificates, representing the easiest to match with only one entry in each record per person, in comparison to marriages and events where each person may have multiple observations. The first step was to remove duplicate rows containing identical information. This process of sub-setting the tables to retain only unique information removed 171 rows from the birth table and 183 from death[1].

Following this the cleaning function was carried out on both tables. This removed white spaces from before or after each string. It also removed blank information and stray punctuation inputted during data entry with no meaning and replaced with “NA” which is standard for the R package “record linkage”.

After the first three rounds there were 1,514 pairs created. Of those 79 were new pairs not existing in the prior database, representing 5.2% of all new IDNR. These new pairs have been extracted for analysis to investigate why these were missed in the initial linkage. Many appear identical, indicating that it is the script cleaning stage which has led to new linkages and in others it is where information is missing and replaced with NA. We however note that to be sure, 1431 cases that has the perfect match are used to create the first step of individual table.

[1] Within the birth and death tables there were some rows for people which were identical within the table rather than between.

3. Specification of standardized variables for meta data

Following Alter and Mandemakers (2014) and metadata file version 4, we have selected several variables provided from COR-2010 which we are planning to release in our first COR IDS version. The relevant variables are marked in yellow noted in the excel file. Most designated variables are present in COR-2010.. Special attention was given for the time stamped information, concerning exact dates, periods and types of missing information (ibid, 21). The complete range of event specific variables is included in this report in an excel format[2].

(Individual level)

- ***Name: last; first***
- ***Birth: date; location***
- ***Stillbirth: date; location***
- ***Multiple birth: number of births***
- ***Legitimacy: yes/no (clarify as they are many items on this)***
- ***Recognition: date; location***
- ***Nationality:***
- ***Title:***
- ***Marriage: date: sequence; location; proclamation date;***
- ***Signature: birth: marriage***
- ***Divorce: date: location***
- ***Occupation: standardized; hisco status; hisco relation***
- ***Death: date: location;***
- ***Event status: alive***
- ***Gender:***
- ***Civil status:***
- ***Age: years; months; weeks; days***
- ***Observation: start; end***

[2] As the file constitutes many notes serving as a process document, the file is not included in the report itself. However, this file can be made available upon request.

- ***Arrival: from; to***

(individual plus)

- ***Relationship: multiple types of familial and non-familial types***
- ***Name of source: population register, birth register, marriage register, funeral register***
- ***Location: Street; house_number; house_name; institution; latitude, longitude; inhabitants; page, volume; sequence number; period***
- ***Relationship to the municipality/locality: birth certificate; marriage certificate; death certificate; baptismal register; marriage register; funeral register; population register***

4. Individual attributes on core variables from output 1

This output is attached to this note in a data format. Individual attributes on core variables from output 1 is included in the individual table. We provide individual attributes of core variables only and indicate consistent linkage (i.e. variable name link) between COR-2010 and COR-test-2017. The test version is attached to this note.

5. State of art on production of indiv-indiv data

The indiv-indiv relationships data serves as a valuable resource in conducting data analysis including social mobility and inequality, social networks and residential mobility etc. However such information requires detailed records of family and non-family networks over both spatial and longitudinal dimensions (Alter & Mandemakers 2017).

IDS in this context requires potentially numerous types of individual-individual information, specifying the exact observation time for each relationship.

Table 2 *Records in the table INDIV_INDIV (excluding timestamp variable)*

Id	Id_D	Id_I_1	Id_I_2	Source	Relation
1	HSN_release_2010.02	1	2	Birth certificate	Wife
2	HSN_release_2010.02	2	1	Population register	Husband
3	HSN_release_2010.02	1	22	Birth certificate	Mother
4	HSN_release_2010.02	22	1	Birth certificate	Child
5	HSN_release_2010.02	2	22	Population register	Father
6	HSN_release_2010.02	22	2	Marriage certificate	Child
7	HSN_release_2010.02	2	23	Population register	Householder
8	HSN_release_2010.02	23	2	Population register	Maid
9	HSN_release_2010.02	2	8493	Population register	Master
10	HSN_release_2010.02	8493	2	Population register	Servant
11	HSN_release_2010.02	823	824	Population register	Sibling
12	HSN_release_2010.02	824	823	Population register	Sibling

Source: Alter and Mandemakers (2014)

Creating indiv-indiv data in COR-IDS requires this information also. However, by consulting with Professor Kees Mandemakers (see meeting notes), we have encountered shortcomings concerning the relationship level information present in the COR-2010 population register. We illustrate this problem first and then provide possible solutions concerning how to convert the original information into the IDS required format.

The first important issue concerns the type of information present in the COR-2010 database and its accuracy. COR-IDS requires a copy of raw materials where the first input file mirrors this exactly. This concerns the recording of the type of event, timing of event and name. COR-2010 not only documents the raw information, but also collects and interprets new information not present in the original material. For instance, all relationship variables regarding the association of individual to the head of household, and also to other household members are added which are interpreted by the data collector (see below). We note that this additional information is a valuable source of information, but introduces potential errors to the data adding serious bias in the data set. In other words, the mixture of raw and obtained new information present in COR-2010 requires additional evaluation to review the quality of the information collected and interpreted by the data collector. This information is also not always complete resulting into high proportion missing.

There are currently at least the following variables present in the COR2010 population register that can be important to the indiv-indiv table.

- (kin) relationship with head of household (relhhh)
- Identifier of head of household (indrhh)
- Identifier of the father (indnrva)
- Identifier of the mother (indnrmo)
- Relationship with other member of the household (rellid1-10): this information is furthermore coded by 60+ types of relationship distinguished by family and non-family relationship.
- Identifier of the family member (indr1-10)

The schema of the relation code is the following (Dutch):

	00: referentiepersoon
0: 0de generatie	01: vader of moeder 02: schoonvader of schoonmoeder 03: stiefvader of stiefmoeder 04: oom of tante 05: voogd 06: grootvader/grootmoeder 07: stiefschoonvader/moeder 08: stiefgrootvader/moeder 09: overgrootvader/moeder
1: 1ste generatie	11: gehuwde partner 12: ongehuwde partner (concubin of concubine) 13: broer of zus 14: schoonbroer of schoonzus 15: stiefbroer of stiefzus 16: halfbroer of halfzus 17: neef of nicht (kind van oom of tante, FR: cousin/e) 18: halfschoonbroer/halfschoonzus

	19: halfneef/nicht (kind van oom of tante)
2: 2de generatie	21: zoon of dochter 22: zoon of dochter uit een vorig huwelijk 23: onwettige zoon of dochter 24: schoonzoon of schoondochter 25 stiefzoon of stiefdochter 27: bevoogd kind 28: neef of nicht (oom/tantezegger, FR: neveu of nièce) 29: halfneef/nicht (oom/tantezegger)
3: 3de generatie	31: kleinzoon of kleindochter 32: stiefkleinzoon of stiefkleindochter 33: schoonkleinzoon of schoonkleindochter 34: achterkleinkind
4: andere familierelatie	40: onbekende familierelatie 41: andere familierelatie
5: geen familierelatie	51: collectief huishouden 52: andere relatie 53: buur (bij aktes) 54: ambtenaar burgerlijke stand (bij aktes) 55: bekende (bij aktes) 56: personeel burgerlijk gasthuis (bij aktes, bv. dienstbode, directeur)
6: onbekende relatie	60: onbekende relatie

Source Van Balen (2007) pp.25-26.

We illustrate here what the nature of the records in COR_2010.

How to get ID mother: First, sort all individual units by identification house. This allows to identify individuals to select to those who have resided one moment in time in the same location. All individual identifier is indicated in IDNR. Number in the household is given where 1 is the head of the household. Identifying the id number of the mother (=2625) can be done by matching the number in the column number in household, Idnrmo and then to IDNR. The

assigned idnr of the mother is created at idmoeder. As one can already see, this process of identifying the id of the mother is complex.

How to identify siblings: rellid equals to 13 for household number 22, 23 and 24. Since 13 is brother or sisters, we assume that these 3 individuals are sisters/brothers and the mother is id 2625. As we interpret this information, we assume that they are biologically related but cannot verify. The remaining rellidX variables and corresponding indrX are great mystery as we are uncertain which relationship they refer to given the fact that in this location, only number of the household 1, 21-24 and 29 exists and all others in between are not recorded. We find the guessing and reinterpreting of the relationship between individuals to be particularly problematic to be used for the creation of indiv-indiv table.

In addition to the above, there are spatial information present in the location table which potentially allows to create the relationship variable when the location of residence is shared. This is the potential information that may be used to create indiv-indiv attribute by replacing the source information of the items noted just above.

- Reference house information concerning
    - Municipality (refgem)
    - Population register (refBR)
    - Quarter number (refwk)
    - Quarter house number (refwkhsnr)
    - Street (refstr)
    - House number (refhsnr)

We consider “workable” solutions to the aforementioned problem given the circumstances: what methods we can apply in obtaining the indiv-indiv table. At the conceptualization stage, we first acknowledge that there exist familial and non-familial relationship to each individual.

The first familial ones are namely determined by intergenerational linkages mostly referring to mothers, fathers and children, and if identifiable aunts, uncles, cousins. We consider that these information can be identified and matched using relevant sources, primarily through certificates.

The second ones refer to the location where non-familial relationship is established when they share the same address for instance. This can be achieved by making use of location table, population register (if possible) and then also with the relevant certificates. For the latter, the obtained location identifier is present (a1, a10, …) (population register, huissamen2) where these sources allow to match with the relevant individuals in the population register. However, if we apply the same to only work with raw inputs and not with the acquired variable present source, we will need to match location with individuals through addresses. This means another linkage based on location variable can be performed by identifying location (addresses) present in the certificates and population register. This step of new direction in our implementation, will however mean that we will first establish a separate table on context, then link between individual and context and lastly determine individual-individual (i.e. individual – context do overlap with this). At this stage, we are rather more confident to identify familial relationship than non-familial ones. Note should be also made that within non-familial relationship there are at least two types: co-residence and non-co residence but based on social network (e.g. witnesses of events).

6. Reference

Alter, G. and K. Mandemakers (2014). The intermediate data structure (IDS) for longitudinal historical microdata, version 4. *Historical life course studies*. 1, 1-26.

Contiero, P., A. Tittarelli, G. Tabliabue, A. Maghini, S. Fabino, P. Crosignan, and R. Tessandori (2005). The Epilink record linkage software presentation and results of linkage test on cancer registry files. *Methods Inf Med*. 44, 1, 66–71.

Fellegi, I. and A. Sunter (1969). A theory for record linkage. *Journal of the American Statistical Association*. 64, 1183–1210.

Haber (1984). AS207: Fitting a general log-linear model. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 33, 3, 358–362.

LONGPOP project document.

Mandemakers, K. and L. Dillon (2004). Best practices with large databases on historical populations. Historical methods: a journal of quantitative and interdisciplinary history. 37, 1, 34-8.

Matthijs and Moreels (2010). The Antwerp COR-database: a unique Flemish source for historical-demographic research. *History of the Family.* 15, 109-115.

Sariyar, M. and A. Borg (2010). The record linkage package: detecting errors in data. Contributed research articles in R Journal vol 2/2, December.

Van Balen, H. (2007) *Constructie van een historisch-demographisch longitudinale database: methodologie van de demographica flandria selecta.* Centrum voor Sociologisch Onderzoek.

Winkler, W.E (1990) String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In *Proceedings of the Section on Survey Research Methods, American Statistical Association.* 354–369, 1990. URL www.amstat.org/sections/srms/proceedings/papers/1990_056.pdf.

Appendix and attachment files

1. **Algorithm of new linkage (syntax)**
2. **Note 1: First meeting with Kees (March)**
3. **Note 2: Kees reaction (April)**
4. **Note 3: Kees-KUL communication April-May 2017**
5. **Note 4: Agenda made for June with F2F meeting with Kees.**
6. **Metadata for COR-IDS (excel file) as process document. Yellow variables is proposed to be included.**
7. **Individual attribute table (excel file)**

1. **Algorithm and syntax of new record linkage**

#Project

#LONGPOP DELIVERABLE 7.1

#Record linkage Birth and Death Certificates COR.

#Libraries to load

```
require(stringr)      #string manipulation

require(RecordLinkage) #linking records

require(gsubfn)       #string manipulation

###Intro###

#Process Steps are:

#1:  Import formatted data

#2:  create perfect linkage

#3:  identify paired records

#4:  identify row numbers of paired records in original data - subset paired data to new dataframe

#5:  remove records identified in step 3 from the original data

#6:  reduce the stringiency of the record matching and create new linkage of remaining records

#7:  identify paired records

#8:  identify row numbers of paired records in original data - subset paired data dataframe created in step 4

#9:  remove records identified in step 3 and 7 from the original data

#10: Repeat steps 6-9 until i) all records are matched or ii) quality of the matches falls below acceptable threshold.

#check for any duplicates:

allDup   = function (value){

  duplicated(value) | duplicated(value, fromLast = TRUE)

}

#cleaning function already carried out as describe in COR_IDSoutput7.1

#####

#1:  Import formatted data

folder = c("C:/Users/u0110986/Dropbox/Data/COR/csv") # change to own foler

folder = c("C:/Users/aarce.LB-ARCELAP/Dropbox (Personal)/Sam/Data/COR/csv")#Change to own folder

birth = read.csv(paste0(folder,"/cleaned_birth.csv"))[,c(1:12,15:17)] #deletes excess names that we wont use (i.e poeple with 7 plus names)

death = read.csv(paste0(folder,"/cleaned_death.csv"))[,c(1:15)]       #deletes excess names that we wont use

birth$row.No = as.factor(row.names(birth))

death$row.No = as.factor(row.names(death))
```

```
#event = read.csv(paste0(folder,"/cleaned_event.csv")) #already had cleaning function


#2:  create perfect linkage
#most stringient linking, has to be a perfect match in the unique
#catagory
a = compare.linkage(birth, death,
                    phonetic = c("unique"),
                    blockfld = c("unique"),
                    strcmp = T,
                    exclude=c(1:14,16)
)
b = emWeights(a, cutoff = 0.8)
perfect.data = getPairs(b)
perfect.data=perfect.data[!apply(perfect.data == "", 1, all),]
perfect.data$quality = as.factor("perfect")
#3:  identify paired records
perfect.data$pair = as.factor(
  paste0(perfect.data$quality,
         "_",
         rep(1:(length(perfect.data$id)/2), each=2)))
paired.data = perfect.data
paired.data = droplevels(paired.data[,c(20,1:19)])
#e.g. pair 17 have identical information but different IDNR numbers (1440 and 21945)
#pair numbers arent consecutive but this doesnt matter, theyre just factors to id
#paired records.
#4:  identify row numbers of paired records in original data - subset paired data to new dataframe
#identify records with a partner, originally I used ID but the rows get re-numbered with every
#matching so im using the row number column which should match the rownames in birth.
perf_birth = droplevels(paired.data[which(paired.data$data.source=="birth"),]$row.No)
perf_death = droplevels(paired.data[which(paired.data$data.source=="death"),]$row.No)
drop_birth = perf_birth
```

```
drop_death = perf_death
#5:  remove records identified in step 3 from the original data
birth_2 = droplevels(birth[!birth$row.No %in% drop_birth, ])
death_2 = droplevels(death[!death$row.No %in% drop_death, ])
rm(perfect.data)
#6: reduce the stringiency of the record matching and create new linkage of remaining records
#repeat with different matching criteria. There are no pairs created if the blocking criteria includes
#more than  the vrnaam 1 and 2. Excluding all other vrnaams
a = compare.linkage(birth_2, death_2,
            phonetic =
c("vrnaam_1","vrnaam_2","vrnaam_3","vrnaam_4","vrnaam_5","famnaam_1"),
            blockfld = c("NISgbplaats","gbdag","gbmaand", "gbjaar","vrnaam_1","vrnaam_2",
                  "famnaam_1"),
            strcmp = T,
            exclude=c(1:3,14:16)
)

b = emWeights(a, cutoff = 0.8)
#
imperfect.data1 = getPairs(b)
imperfect.data1 = imperfect.data1[!apply(imperfect.data1 == "", 1, all),]
imperfect.data1$quality =as.factor("a1")
imperfect.data1$pair = as.factor(
  paste0(imperfect.data1$quality,
      "_",
      rep(1:(length(imperfect.data1$id)/2), each=2)))
#identify records with a partner:
imperf_birth1 = droplevels(imperfect.data1[which(imperfect.data1$data.source=="birth"),]$row.No)
imperf_death1 =
droplevels(imperfect.data1[which(imperfect.data1$data.source=="death"),]$row.No)
#difficult to join list of factors. Need to convert to character, then join, then return to factor
drop_birth = as.factor(c(as.character(drop_birth),as.character(imperf_birth1)))
```

```
drop_death = as.factor(c(as.character(drop_death),as.character(imperf_death1)))
#join the paired values from this round to the paired.data file
paired.data = rbind(paired.data, imperfect.data1)
rm(imperfect.data1)
birth_3 = droplevels(birth[!birth$row.No %in% drop_birth, ])
death_3 = droplevels(death[!death$row.No %in% drop_death, ])
#round 3
#What is the least informative criteria, i.e. the one where we can accept some level of error?
#I think probably the "NISgbplaats" code or the "gbdag". Trying to delete "NISgbplaats" 1st
#
a = compare.linkage(birth_3, death_3,
                phonetic =
c("vrnaam_1","vrnaam_2","vrnaam_3","vrnaam_4","vrnaam_5","famnaam_1"),
                blockfld = c("gbdag","gbmaand", "gbjaar","vrnaam_1","vrnaam_2",
                        "famnaam_1"),
                strcmp = T,
                exclude=c(1:3,14:16)
)
b = emWeights(a, cutoff = 0.8)
#
imperfect.data2 = getPairs(b)
imperfect.data2 = imperfect.data2[!apply(imperfect.data2 == "", 1, all),]
imperfect.data2$quality =as.factor("a2")
imperfect.data2$pair = as.factor(
  paste0(imperfect.data2$quality,
        rep(1:(length(imperfect.data2$id)/2), each=2)))
imperfect.data2      = assign_pairs(imperfect.data2)
#identify records with a partner:
imperf_birth2 = droplevels(imperfect.data2[which(imperfect.data2$data.source=="birth"),]$row.No)
imperf_death2 =
droplevels(imperfect.data2[which(imperfect.data2$data.source=="death"),]$row.No)
```

```
#difficult to join list of factors. Need to convert to character, join, then return to factor
drop_birth = as.factor(c(as.character(drop_birth),as.character(imperf_birth2)))
drop_death = as.factor(c(as.character(drop_death),as.character(imperf_death2)))

#join the paired values from this round to the paired.data file

paired.data = rbind(paired.data, imperfect.data2)
rm(imperfect.data2)



#turning quality into numeric
summary(paired.data$quality)
paired.data$quality2 <- as.numeric(paired.data$quality)
summary(paired.data$quality2)
describe(paired.data$quality2)
paired.data$quality2 <- as.factor(paired.data$quality2)
paired.data$IDNR2 <- paste(paired.data$pair ,paired.data$quality2)

describe(paired.data$IDNR)
describe(paired.data$IDNR2)
#which pairs are wrong?
 = duplicated(paired.data$IDNR)|duplicated(paired.data$IDNR, fromLast = TRUE)
summary(paired.data$dup)
summary()
# 79 pairs with different old IDNR
#extract these to view
newid <- paired.data[ which(paired.data$dup=='FALSE'),
write.csv(paired.data, "C:/Users/u0110986/Dropbox/Data/COR/pairs.csv", row.names=FALSE)
write.csv(newid, "C:/Users/u0110986/Dropbox/Data/COR/newid", row.names=FALSE)
write.csv(paired.data, "C:/Users/u0110986/Dropbox/Data/COR/pairs.csv", row.names=FALSE)
```