Aldo Benini

Efficient linking of lists in humanitarian data management

MATI YA KUDHIBITI
UKIMWI
KAMATI YA MAJI
WAHUDUMU WA MAJI
BARAZA LA KUDHIBITI UNYANYASAJI KIJINSIA
KITENGO CHA REDESO
RUSOHOKO
OFISI YA REDESO RUSOHOKO
MAJINA YA WAJUMBE WA KAMATI YA UZUIAJI WA UNYANYASAJI WA KIJINSIA RUSOHOKO-KIBONDO
TCRS
FADILI TABIA

Two HIV Committee volunteers attached to the Rusohoko Village Council, in Kibondo District of western Tanzania, sit beneath a variety of wall posters, some of which are lists assigning councilors or volunteers to particular sub-villages and functions. The council as well as these volunteers are supported by the Tanganyika Christian Refugee Service (TCRS), which is active both in the local village communities and in the nearby camp of refugees from Burundi.

Just a few meters from the office entrance, a large signboard draws attention to the "Sexual and Gender Based Violence Office (SGBV) at Rusohoko Village, Funded by UNHCR and Constructed by REDESO", another NGO. All these agencies produce, and occasionally exchange, lists of various kinds.

Summary

Relief workers sometimes have to match two or more lists of persons (food aid recipients, camp populations, missing persons, patients, etc.) or localities (villages of origin; populated places in two administrative gazetteers). The identifying information (name, address, document numbers) may be held in spreadsheets or databases, but may defy immediate matching, notably because of spelling differences. Automated record linkage procedures can speed up the process greatly while manual verification of dubious cases remains important. With lists obtained from a community empowerment program in Tanzania, I demonstrate how the linkage works, using one method in a popular spreadsheet application, and another in a statistical program.

Contents

Summary 1

Acknowledgments 3

Introduction 4

An experiment with lists from Tanzania 5

The generic two-list case 6

Some of the mechanics involved 7

Two procedures and some of their notable features 9

“reclink” for STATA 9

“FuzzyVLookup” for MS Excel 10

Results of matching the tables from Tanzania 11

The baseline and progress assessment tables 11

Overall distribution of matching scores 11

“reclink” without variable weighting 12

“reclink” with weighted variables 13

“FuzzyVLookup” with concatenated name and village 13

Lessons from this experiment 14

Matching on variables other than strings 14

Conclusion 15

Appendices 16

The STATA procedure “reclink” 16

Log file segment for a “reclink” job 18

Inspecting several match candidates with “FuzzyVLookup” 21

Matching scores in “reclink” and “FuzzyVLookup” compared 23

References 24

About the author 26

Acknowledgments

Michael Blasnik, M. Blasnik & Associates, Boston, wrote the STATA "reclink" procedure, corrected several of my earlier assumptions and gave permission to reproduce the entire "reclink" help documentation in the appendix. An unknown Excel forum participant wrote the "FuzzyVLookup" and associated functions.

Adam Jackson, Everready Nkya and Samwel Chago, TCRS, Dar-es-Salaam and Kibondo, provided the data that I use in a table linkage experiment.

These individuals contributed useful comments and/or circulated an earlier draft widely: Charles Conley, Olivier Cottray, Petri Nevalainen, Bernard Chedid, iMMAP, Fairfax, Nairobi, Amman, and Kabul; Firoz Verjee, George Washington University; Paul von Bünau, Technical University, Berlin; Andrew Alspach, UNOCHA, Geneva; Daniel Eriksson, UNDP, Amman; Kristian Gleditsch, University of Essex, Colchester; Henrik Urdal, PRIO, Oslo.

Introduction

In 1990-91, I led the International Committee of the Red Cross (ICRC) team in Wau, then a besieged city in southern Sudan, in the war opposing government and SPLA forces. The ICRC distributed monthly food rations in four camps for displaced persons and in a number of institutions. Occasionally, distributions were held for large segments of the town population that assessment teams identified as needy. Individuals and families registered in camps (although not necessarily living there) and those selected by a government relief commission or the local branch of the Sudanese Red Crescent were communicated to the ICRC office by way of lists[1]. Those registered in the camps were issued ration cards. Besides the commodity management, the relief team would spend the major part of their time verifying, updating and, in frequent disputes with stake-holders, adjudicating beneficiary lists. Often, this would involve comparing lists - older and newer lists of the ICRC's own, or the ICRC's vs. some other requestor's. The process was done entirely manually, alternating between batch jobs around a large office table and itinerant verification by field officers. There were no computers for list processing although I simulated alternative food distribution scenarios and their predicted mortality reduction outcomes in models run on a primitive (by today's standards) Toshiba 286 laptop machine (Benini 1991).

Humanitarian information management has since become a discipline in its own right (United Nations 2008); computing skills and power pervade also the micro-management of relief. Nevertheless, the comparison of lists (of persons entitled, claiming, approved and serviced) remains a practical challenge (Telford 1997). Sociologically, lists are a fascinating mixture of cognitive and normative expectations – claims that somebody exists by a particular description, claims that he/she is entitled to something – and therefore an ideal object of group contention. For example, until recently there were an estimated 30,000 displaced persons living in a large number of official and make-shift camps in the capital of East Timor, Dili. Despite all efforts to verify lists of actual camp dwellers, the number of persons listed for food distribution has remained as high as 70,000[2].

Technically, the choice of descriptors and the quality of individual records determine what can be achieved with a list. This includes the ability to link it to other lists or, more generally, to tables with records of a common interest. Lists, of course, will continue to be compared manually, by individual workers or through organized division of labor among team members or even larger and partly remote units. But technologies developed for the automated linking of table records can greatly accelerate matching and comparison. They help relief workers to focus attention on dubious, duplicate, unmatched or otherwise interesting cases, and to integrate the comparison work with other analytic tasks. The rest of this note is devoted to technicalities of rapid linking, not to the

[1] Obviously, many recipients, in the camps and in the town, were personally known to ICRC workers, notably to the field officers, who spoke their language. As members of lists, however, they assumed a variant of the double identity that clients of bureaucracies must step into, as living persons moving around in their bodies, and as files or cases stuck in paper or hard drives.

[2] Conversation with an UNOCHA official, Dili, May 2008.

substantive contents of lists, or to the social power implications that the act of listing populations and the technologies of table management carry – all worthwhile topics in themselves[3].

An experiment with lists from Tanzania

In summer 2007, I spent time in three of the western districts of Tanzania, visiting local units of the Tanganyika Christian Refugee Service (TCRS) that were conducting parts of the TCRS' flagship Community Empowerment Program (Benini 2008). A hallmark of its management was that some of the district administrators took strong initiative to evaluate the impact of their programs. Their technical and professional infrastructure, however, was very limited. The TCRS head office in Dar-es-Salaam, while giving guidance for baseline surveys and for phasing-out policies, was not in a position to effectively support district units with survey design or data management.

In Kibondo, the Project Coordinator shared copies of spreadsheets with data from a recent sample survey of the economic progress of CEP participant households (Nkya and Chago 2007). I was told that baseline questionnaires, filled in 2005 on all households of the participant villages, were being kept in a storeroom, and that a former expatriate advisor had discouraged transfer of (some or all of) that information to spreadsheets. At my request, a monitoring person from the head office went to work in Kibondo for a short period of time in early 2008, to help project staff locate baseline questionnaires from 2007 sample villages and capture some of the information. After creating spreadsheets for several hundred baseline records from five villages, he was called back; the linkage to the 2007 survey could not be performed, and comparative data sadly was unavailable in time for my report-writing. The spreadsheets with year 2005 data were later shared, without links to the 2007 data (Nkya 2008). A brief visual inspection of the tables for both years convinced me that the trouble of matching records, if done manually, was indeed forbidding. In particular, the spelling of names was widely different between the tables, and so were the 2007 sample sizes and the number of baseline questionnaires entered for any of the five villages.

The situation changed when I found out about a procedure that automates record linkage between tables for which identifying information consists of string variables (names,

[3] The Sphere handbook treats lists as an element of the registration process, in the Food Aid chapter, with considerations that are primarily social:

> *"Registration: formal registration of households receiving food aid should be carried out as soon as is feasible, and updated as necessary. Lists developed by local authorities and community-generated family lists may be useful, and the involvement of women from the affected population in this process is to be encouraged. Women should have the right to be registered in their own names if they wish. Care should be taken to ensure that female or adolescent-headed households and other vulnerable individuals are not omitted from distribution lists. If registration is not possible in the initial stages of the emergency, it should be completed as soon as the situation has stabilised. This is especially important when food aid may be required for lengthy periods"* (Sphere Project 2004: 169).

As far as I can see, there are no technical standards for lists.

addresses, document numbers, etc.) with spelling differences. Michael Blasnik's "reclink" (Blasnik 2007) has the advantage of working within a statistical package (STATA) ideally suited to analyze survey or census data, including the special case of tables of relief beneficiaries or participants in community development programs.

However, not everyone needs or wants to work with this particular application. The majority of humanitarian data managers and relief logisticians are familiar with spreadsheets, databases and, increasingly, Geographical Information System (GIS) applications; few of them use econometric software such as STATA. Luckily, an anonymous Excel list member has contributed a small suite of functions that cover some of the features of "reclink" (and a few that "reclink" does not provide). In addition, interested users can download a small number of freeware programs that offer similar functionality, such as the CDC's "Link Plus" (CDC 2008).

In this paper, I will present "reclink" for STATA as well as "FuzzyVLookup" for Excel (Anonymous 2006). My choice of "reclink" is due to some of its superior features and because of Blasnik's excellent presentation. "FuzzyVLookup", together with its associated "FuzzyPercent", is chosen because of the wide use of this spreadsheet software; many Excel users are familiar with the deterministic lookup function "vlookup". While gladly advertising "reclink" and "FuzzyVLookup", I will present the linkage challenge generically[4]. Further below, I will present some of their notable features separately and discuss their performance in linking the Tanzania lists.

The generic two-list case

We assume two tables, with a number of variables appropriate for case identification and then for linkage. In this example (see tables below), "Name" and "Location" are used, with first and family names and suffices in one field (they could be handled in separate fields) and only the city name as an address element given here. This, realistically, is often the case of beneficiary lists, with relief agencies not completely understanding local naming conventions, and with only a vague indication of an address or place of origin or group affiliation. For convenience, records are arranged in pre-matched pairs, with differences highlighted yellow. Note that the running number of records is accidental to the personal information here; "reclink" (and presumably the freeware applications), however, require some kind of a unique identifier, if only in the shape of a simple record numbering.

It is obvious that only records #1 and 2 are perfect matches. Cases #3 and 4 are spelling-different; 5 and 6 may be genuinely different individuals (with a spelling difference for

[4] I have not investigated "LinkPlus", although a reader pointed out its attractive phonetic matching algorithm, in addition to variable matching, a feature particularly helpful when transliteration presents major challenges. Similarly, I was told, but have not verified, that Microsoft Access has fuzzy matching facilities. One of the teams researching the Srebrenica massacre used them in combining lists of missing persons from the ICRC and from the Physicians for Human Rights, which were then matched with voter lists and census files (Brunborg, Lyngstad et al. 2003: ; Urdal 2008). Schnell and Bender (2004) developed "Matching Tool-Box" (MTB), a Java-based program using the STATA data format, but I have not found out whether this program is available freely. Industrial-strength linkage operations (such as those run by census bureaus) are discussed in a data quality perspective in Herzog et al. (2007).

the location, and diacritics in the name, of #6). #7 may reflect different individuals or an error in the location. In #8, only local knowledge can determine whether "Uptown" is a spelling mistake or a different location in its own right. #9 does not seem to have a partner in the second list.

RunNo1	Name	Location
1	Alec Miller	Middletown
2	Alec Muller	Middletown
3	Bobby Gonzales	Middletown
4	Bob Glenn	Middletown
5	Bob Glenfield	Middletown
6	Maria Guzmán	Middletown
7	Harry Glenn	Middletown
8	Angela Glenn	Uppertown
9	Alec Muller II	Uppertown

RunNo2	Name	Location
1	Alec Miller	Middletown
2	Alec Muller	Middletown
3	Bobbi Gonzales	Middletown
4	Robert Glenn	Middletown
5	Bob Glennfild	Middletown
6	Maria-Angela Guzman	Midletown
7	Harry Glenn	Uppertown
8	Angela Glenn	Uptown

Lists as small as these can be efficiently matched by a person with modest spreadsheet skills (e.g. for sorting) and with the substantive knowledge to decide dubious cases. For larger tables and lists, an automated procedure is desirable, but the basic constellations are the same: perfect matches, cases for which the assumption of a spelling difference is reasonable, cases for which additional knowledge is needed to decide a suggestive match, and unmatched cases.

Some of the mechanics involved

"reclink" and "FuzzyVLookup" exploit probabilistic theories of record matching, going back 40 years (Fellegi and Sunter 1969) to the intuition that if two records are a true match the probability that they exactly match on some identifying variable (e.g., a person's name) is greater than in the case of distinct entities. Later developments have essentially been to admit imprecise matches for strings, e.g. Benini and Benigni (He of "Life is Beautiful"!), and to define and calculate measures for the degree of matching.

The measure exploited by "reclink" is the proportion of two-character sub-strings that the strings have in common:

Benini	Benigni	Common
Be	Be	1
en	en	1
ni	ni	1
	ig	0
	gn	0
in		0
ni	ni	1
5	**6**	**4**

which works out, as 4/max(5, 6) = 0.67[5]. This can be used, in combination with the proportions of common strings in the other selected variables, to form a matching score. The record to be matched in is then selected as the one with the highest matching score, with perfect matches scoring 1. An additional precaution that some linkage programs, including “reclink”, employ is known as “Or-blocking”. It means that only those records are selected that match exactly at least on one of the identifying variables.

When we link table 2 to table 1 using “reclink” , we obtain:

Runno1	Name1	Name2	Location1	Location2	Score	Runno2
1	Alec Miller	Alec Miller	Middletown	Middletown	1	1
2	Alec Muller	Alec Muller	Middletown	Middletown	1	2
3	Bobby Gonzales	Bobbi Gonzales	Middletown	Middletown	0.9869	3
4	Bob Glenn	Bob Glennfild	Middletown	Middletown	0.9772	5
5	Bob Glenfield	Bob Glennfild	Middletown	Middletown	0.9845	5
6	Maria Guzmán	Maria-Angela Guzman	Middletown	Midletown	0.8979	6
7	Harry Glenn	Robert Glenn	Middletown	Middletown	0.5901	4
8	Angela Glenn	Angela Glenn	Uppertown	Uptown	0.9018	8
9	Alec Muller II	Harry Glenn	Uppertown	Uppertown	0.5038	7

Three of the nine matches are incorrect. In #4, the procedure matched Bob Glenn with Bob Glennfild, rather than with the expected Robert Glenn because the first option produces a higher proportion of common two-character strings than the second would.

The mismatches in #7 and 9 are artificial. They occurred by forcing some match at all costs, i.e. with a lower matching score than the default threshold. Harry Glenn of Middletown, in #7, was matched with Robert Glenn, also of Middletown, rather than with Harry Glenn of Uppertown. Alec Muller II, in #9, was paired, more or less arbitrarily, with the one denizen of Uppertown; for the location contributed more common strings than the name would.

Note two things. Some records from the second table have been used several times. All correct matches had a score > 0.85. I found that setting the minimum score to 0.90 was advantageous when working in “reclink” with only two identifying variables, name and location (the default threshold is 0.60). Second, be aware of case-sensitivity; it may be necessary to convert all identifying variables in both tables to lower case before the analysis, perhaps using copies if the original spellings will be needed later on.

Last, but not least, the author of “reclink” sounds this warning:

> *“In general, record linkage methods are imperfect and results should be manually reviewed, especially for observations with lower matching scores. It is not uncommon to try several runs with a variety of weights, Or-block options, and derived variables to increase the accuracy of the linkage.”*

[5] Technically the two-string characters are known as “bigrams” (and the three letter string as a “trigram”). They play an important role in language pattern analysis (Manning and Schütze 1999).

The need for manual inspection, sometimes known as “clerical review” (Machado and Hill 2004: 915), is echoed widely in the literature, but, of course, the primary rationale for automated linkage methods is to reduce or eliminate it altogether.

Two procedures and some of their notable features

Apart from being embedded in two different applications – STATA and Excel -, these two procedures offer distinct strengths. Users may want to choose one or the other opportunistically, in view of the task at hand or of the ability of other participants to further process intermediate results.

“reclink” for STATA

For STATA users, the complete syntax and definitions of options are given in the appendix. Of general interest here is its ability to

- Match on several variables
- Assign different weights to matches (and, separately, to mismatches) on each of the matching variables
- Offer Or-blocking (i.e., allow a match only if records match exactly at least on one of the Or-blocked variables) and, simultaneously if desired, And-blocking (must match exactly on all variables in the And-block)
- Exclude a subset of records previously matched from entering the current procedure, either to speed up calculation or to rerun it for the unmatched records, using different matching criteria
- Set the minimum matching score needed to admit a match of two records, thus changing the balance between false positives (declared matches that are incorrect) and false negatives (a correct match is excluded).

When “reclink” finds several best candidate matches with identical scores (a “tied” first rank), it creates a record for each of them. The specific values of the identifying variables are imported for each candidate record. This facilitates speedy manual inspection (Blasnik 2008), but the consequence is that the record identifier of the receiving table is no longer unique. If a unique identifier is required, records beyond the first candidate match will have to be eliminated manually or through a general procedure[6].

Blasnik also wrote a STATA routine for the Soundex algorithm, a variant of phonetic encoding (Blasnik 2001: ; Wikipedia 2008). Soundex codes can thus be computed and used as an additional identifying variable.

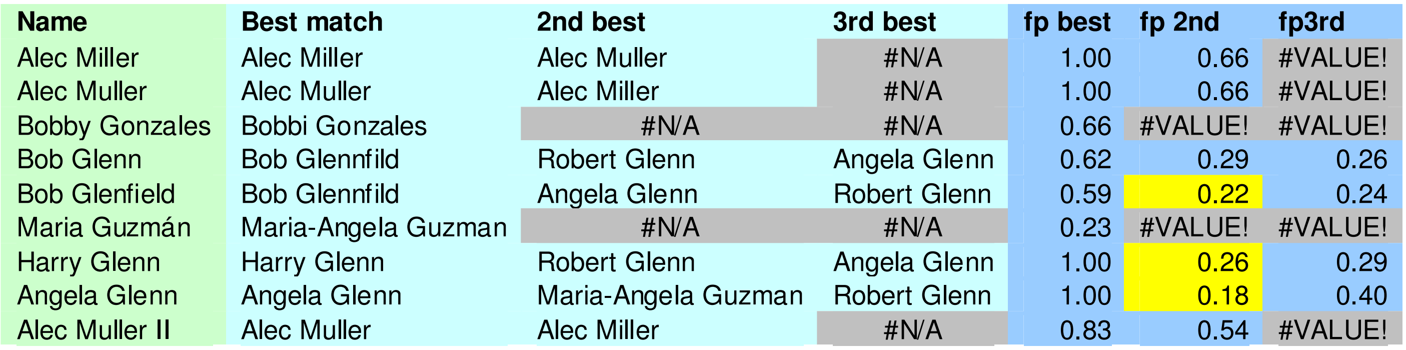
[6] Multiple best candidate creation is a feature both comfortable and dangerous. Another risk for the novice is that substantive variables are imported wholesale, which saves work. However, those that have identical names in both tables will appear in the merged table only once (in the way the STATA command “merge” handles them). Thus, if the master and using table variables of same name are to remain distinct, one set will have to be renamed prior to running “reclink”. The price of comfort is eternal vigilance.

“FuzzyVLookup” for MS Excel

“FuzzyVLookup” has been contributed by a participant (Anonymous 2006) of a Microsoft Excel forum that appears to be part of a commercial site. The code and some explanations, however, can be downloaded freely, without signing up to the forum. Besides fitting with a widely used and extremely flexible spreadsheet application, FuzzyVLookup has a few more attractions to recommend it:

- Its syntax is an extension of a well-known workhorse of Excel’s data management capabilities, the function “vlookup”.
- A ranking option makes it easy to present the best, second-best, etc. match side-by-side horizontally for quick visual inspection.
- It comes as part of a mini-suite of fuzzy linkage functions. The other valuable member is the function “FuzzyPercent”, which returns the degree of match between two string variables. “FuzzyPercent” runs separately from “FuzzyVLookup” and can be used in contexts other than lookup and linkage[7].
- Both functions offer the choice of different matching algorithms.

Like “reclink”, “FuzzyVLookup” lets the user set a minimum matching score below which a match is not considered to occur. This table assembles, for each of the names in the leftmost column, the most highly ranked matches above a score of 0.1.



Name	Best match	2nd best	3rd best	fp best	fp 2nd	fp3rd
Alec Miller	Alec Miller	Alec Muller	#N/A	1.00	0.66	#VALUE!
Alec Muller	Alec Muller	Alec Miller	#N/A	1.00	0.66	#VALUE!
Bobby Gonzales	Bobbi Gonzales	#N/A	#N/A	0.66	#VALUE!	#VALUE!
Bob Glenn	Bob Glennfild	Robert Glenn	Angela Glenn	0.62	0.29	0.26
Bob Glenfield	Bob Glennfild	Angela Glenn	Robert Glenn	0.59	0.22	0.24
Maria Guzmán	Maria-Angela Guzman	#N/A	#N/A	0.23	#VALUE!	#VALUE!
Harry Glenn	Harry Glenn	Robert Glenn	Angela Glenn	1.00	0.26	0.29
Angela Glenn	Angela Glenn	Maria-Angela Guzman	Robert Glenn	1.00	0.18	0.40
Alec Muller II	Alec Muller	Alec Miller	#N/A	0.83	0.54	#VALUE!

Some of the technicalities are further discussed in the appendix. The documentation is not entirely satisfactory (notably on error types and on the use of multiple identifying variables through a concatenating option); and I have encountered inconsistencies in less-than-best matches[8]. And, as the author warns, Excel is slow to execute the code, particularly when the tables are large.

Nevertheless, this is a valuable tool, requiring only a minor learning effort of Excel users who wish to try it out, and thus a privileged candidate tool for the humanitarian community.

[7] The horizontal lookup equivalent, “FuzzyHLookup”, is of lesser interest, given the way most data tables are arranged.

[8] See yellow cells in table above – for these matches the score (fp), computed by “FuzzyPercent” does not decrease monotonously.

Results of matching the tables from Tanzania

The baseline and progress assessment tables

Returning to our experiment from Tanzania, the 2007 assessments of household progress were held in separate worksheets, one per village. For the five villages for which an attempt was made to create comparisons with the baseline, information was available on 893 participants. Later, the visiting monitor helped the Kibondo project staff enter information from 571 baseline questionnaires, again in a separate worksheet for each of the five villages.

Village, 2007 spelling	Participants surveyed in 2007	Village, different spelling in 2005	Baseline 2005 questionnaires entered
Kiduduye	250		100
Kigendeka	160		100
Kumbanga	115		170
Kumuhasha	189		100
Mkabuye	179	Mukabuye	101
Total	**893**		**571**

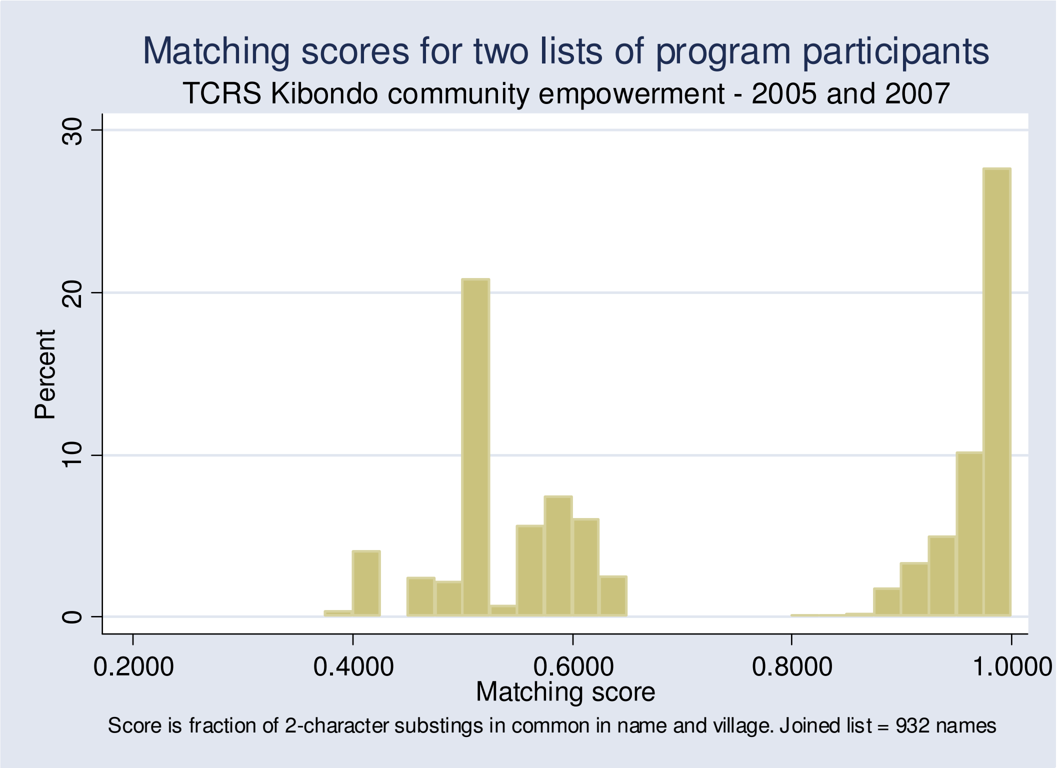
I am not concerned with sampling questions here, but solely with record linkage. Note that, since the data for each village was kept separately, the linkage would in real life be attempted on a village basis. I stacked the records for all five villages in a common table for the sake of demonstration.

Overall distribution of matching scores

It makes a difference whether one uses the 2007 table as the main table, and the 2005 table as the lookup table from which records are probabilistically matched to 2007, or whether we proceed the other way round.

At first, I used the 2007 table as the main and set a very low minimum matching score (0.20), in order to force a match to each of the year 2007 records. This produced a joint table with 932 records, i.e. one that duplicated 932 – 893 = 39 records to accommodate records from 2005 with identical matching scores. In practice, one would not want to operate with such a low standard, but it allows us to see the distribution of matching scores when "all will get prizes". Clearly, as the graph on the next page shows, the scores are clustered around in the 0.4 – 0.65 and in the 0.90 – 1 ranges. The first cluster is filled with matches for which only the village names agreed, the second with those for whom the village as well as the interviewee names agreed strongly or even perfectly.

Two insights follow, at least in this constellation. First, a high (>0.90) minimum matching score will be needed. Second, it is advantageous to weight the identifying variables, which here means to give greater weight to the person's name, as opposed to the village. I repeated the experiment, first by raising the minimum score, without variable weighting, ten in a second round, by introducing such weights.



“reclink” without variable weighting

Repeating the linkage with a minimum matching score of 0.90, 426 records were matched, 104 exactly (no spelling differences). 467 records in the 2007 table remained unmatched. 4 of the matched records were duplicated because more than one candidate being brought in from the 2005 list produced the same highest score[9].

The other way round, using the same threshold, out of the 571 baseline questionnaires from 2005, 408 were matched with 2007 assessments. 105 of the matches were exact. 163 questionnaires could not be matched. 2 questionnaires attracted duplicates of equal matching scores:

RecID2005	Name 2005	Name 2007	RecID2007	Village	Matching score
142	julia wilbard	doralia wilbart	789	Kumbanga	0.9423
142	julia wilbard	merania wilbard	788	Kumbanga	0.9423
227	jaclina antony	jekelina antoni	561	Kiduduye	0.9309
227	jaclina antony	katelina antoni	558	Kiduduye	0.9309

The examples are instructive. ID2005 #142, Julia Wilbard, was mismatched in both instances, despite a high score of 0.94, reinforcing the case for visual inspection of of imperfect matches. #227, Jaclina Antony, probably is the same person as Jekelina Antoni

[9] Code for identifying and listing duplicates in STATA is given on page 18.

in 2007. However, Katelina and Jekelina produce the same proportion of matching two-character strings, thus the same score.

“reclink” with weighted variables

I then repeated the last procedure, the linking of 2007 assessment records to 2005 baseline records, using weights. I gave names five times the weight of villages. The weighted procedure found the same 105 perfect matches. However, the incomplete matches diminished because, with the higher weighting of differences in name spellings, more cases were pushed below the 0.90 minimum matching score.

Code and examples of the 44 additional exclusions are shown in the appendix. Substantively, the improvement was spurious. There are a number of false negatives, meaning that matches that in all likelihood were correct now are excluded because of name spelling standards that are too high.

More experimentation may be warranted, perhaps by at first fixing spelling differences in village names (which are few and easily compared), then requiring exact matches on village, slightly lower minimum scores and more intensive manual inspection of proposed matches.

“FuzzyVLookup” with concatenated name and village

I repeated the experiment in a spreadsheet, using “FuzzyVLookup” and setting a very low minimum score in order to force matches for all cases and compare them to the “reclink” results. In the event, a score of 0.10 produced a match for all but four of the 571 records in the 2005 baseline table.

For a realistic comparison, I took those records for which the unweighted “reclink” procedure, with a high 0.90 minimum score, had produced a match. There are 408 such cases. In 380 (93 percent), “FuzzyVLookup” makes the exactly same match as “reclink” does.

Looking at cases for which the two procedures produce different matches, a few examples are instructive:

Recno2005	**Name in 2005**	**Match 2007:** ***“reclink”***	**Match 2007:** ***“FuzzyVLookup”***	***Comment***
15	bernadeta kibaya	bernadetha kisaya	bernadeta anthon	*reclink plausible*
47	anastazia kasela	anastazia konsela	anastazia kanyimba	*Undecidable*
86	mashinga fabarushe	mashinga sejuru	mashinga kibaba	*Both implausible*
90	ruzalia filbert	yulia wilbert	ruzalia busumihyo	*Both implausible*
191	elizabeth	elizabet tadeo	elizabeth saidi	*Undecidable*
192	bernadeta yohana	belenadeta yohana	bernadeta anthon	*reclink plausible*
227	jaclina antony	jekelina antoni	katelina antoni	*Both plausible*
480	veronika chza	veronica chiza	veronika chiza	*FuzzyVLookup plausible*

Thus, several constellations appear. Reviewing all the 22 divergent cases reveals that the "reclink" results were more plausible in 14 cases, the "FuzzyVLookup" results in 2, with the remaining cases showing undecidable, or equally plausible or implausible matches.

In normal work situations, when results from another application are not available, where should the minimum matching score be set in "FuzzyVLookup"? This is a difficult question because this function does not return scores directly. When its associate "FuzzyPercent" is called up for the purpose, a large number of good matches (those identical with the "reclink" results) produce low numbers. Half of the values returned by "FuzzyPercent" are below 0.73. The lowest value in this set is 0.259.

This may recommend a two-step procedure for "FuzzyVLookup" users. For a first round, the minimum score may be set relatively high, say, to 0.80. These matches may all turn out perfect or highly plausible. New tables could then be created, through a simple sort operation, excluding the cases already successfully matched. The second round, with the new tables and using a low threshold, can be used to produce several candidates for each match. These can then be inspected visually. At the end, both tables are merged. This is less convenient than working in "reclink", with its directly documented and sharply discriminating thresholds, but it is still an enormous efficiency gain over purely manual or deterministic linking attempts.

Lessons from this experiment

What should be learned from this example for general purposes? The proportion of unmatched records is specific to this data situation and holds no general lessons. However, the low proportions of *exact* matches within all matches may be fairly typical in situations where only string variables serve as identifiers, and even more so where, as in Swahili-speaking Tanzania, names may be written in different transliterations. Also, exact matches may be all the rarer where orthographic and typing skills are low in enumerator and data entry personnel. In either direction, of all the matched records, only a quarter were exact matches.

The consequence is brutally clear. Linking beneficiary lists using a deterministic process, such as Excel's "vlookup" function (without "Fuzzy"!), will give results for a fraction of records only, leaving most of the work to be done tediously by resorting tables and visual inspection. Probabilistic methods like "reclink" or "FuzzyVLookup" produce a first cut that covers most cases. The results still need to be inspected visually. But the proportions of incorrect, dubious and unmatched cases are much lower, allowing workers to focus on smaller numbers that need individual re-evaluation and re-matching.

Matching on variables other than strings

The two procedures that I have presented in some detail, as we have seen, link records based on matching string variables. These may include text as well as numbers or dates turned into strings. However, not all data situations that call for record linkage are best managed using string variables. A substantively justified linkage may be expressed through relationships other than personal, group or location names. In humanitarian action, spatial relationships are particularly important because of the physical nature of many of the needs, of the resources to meet them, and of the logistical and access challenges that separate the victims from the providers.

Therefore, linkage based on spatial relationships may be productive, even if the entities so linked are of different physical or ontological content. Nearest neighbor relationships and others based on particular spatial search operators may be of particular interest. For example, in an analysis of a rapid explosive-remnants-of-war (ERW) survey in Iraq, Benini and Conley (2007) linked survey points to communities previously surveyed under the 2003 United Nations Rapid Assessment Process by forming Thiessen polygons around the community center coordinates. Since several small hamlets each may have received a separate ERW survey within the hypothetical territory of a community (approximated by the polygons), name matching would have been pointless[10].

Numeric relationships other than spatially defined ones may also be of interest for certain humanitarian analysis concerns, such as in case-control studies of nutritional situations or of disaster impacts, given exposure and pre-existing socio-economic profiles. However, these situations are too distant from the normal bread-and-butter list matching and record linkage to warrant further discussion here[11].

Conclusion

Matching lists is a frequent challenge in humanitarian data management. In many, if not most situations, common and unique record identifiers across lists and tables are absent. Records may be linked using descriptor variables such as individual, family, group and location names. Because of spelling and other differences, exact matches, produced by deterministic methods, may work for a small proportion of the records only.

Probabilistic methods greatly accelerate the matching job. The software offered for automated record linkage ranges from freeware to $200,000-a-licence Rolls-Royce applications. I reviewed two affordable ones. In automobile lingo, STATA's "reclink" may qualify as a fast and comfortable Chevrolet, with some behaviors that demand the driver's heightened attention. Excel's "FuzzyVLookup" is the Volkswagen that most people can drive, slower and with fewer bells and whistles. But it will get you there.

Despite the lure of automation, visual inspection of the less-than-perfect among the suggested matches and local resolution of dubious and resistant cases remain necessary for good results. An experiment linking two lists of participants in a community empowerment program in Tanzania, established from surveys two years apart, revealed

[10] A reader drew attention to the so-called "p-codes", a standardization device promoted, particularly by UNOCHA Humanitarian Information Centers, for the consistency of administrative gazetteers (UNOCHA 2001). However, p-codes (= place codes) are assigned to gazetteer elements that have already been thoroughly reviewed, including through linking several initial gazetteers and place name databases – they are not a fuzzy record linkage technique. In fact, part of the rationale for creating p-codes is to have standardized unique identifiers for later deterministic linkage of tables (including map theme attribute tables).

[11] STATA users interested in the management of such situations may explore the procedure "nnmatch", using the option "keep", in combination with the procedure "expand".

that only one quarter of the records matched on participant name and village were exact matches.

Therefore, quality improvements will not primarily result from further statistical refinement, but rather from those investments in instrument design, training and supervision that are key to minimizing measurement error. Still, the methods that I have presented in this paper will reduce the tedium of data management and will thus ultimately support faster and better decision-making. They have a legitimate place in the toolbox of the humanitarian data manager, and they should be introduced in training courses for United Nations and NGO program monitoring and evaluation personnel.

Appendices

The STATA procedure "reclink"

The help function for "reclink" (Author: Michael Blasnik)

Record Linkage

reclink varlist using filename , idmaster(varname) idusing(varname) gen(newvarname) [wmatch(match weight list) wnomatch(non-match weight list) orblock(varlist) required(varlist) exactstr(varlist) exclude(filename) _merge(newvarname) uvarlist(varlist) uprefix(text) minscore(#) minbigram(#)

Description

reclink uses record linkage methods to match observations between two datasets where no perfect key fields exist -- essentially a fuzzy merge. reclink allows for user-defined matching and non-matching weights for each variable and employs a bigram string comparator to assess imperfect string matches.

The master and using datasets must each have a variable that uniquely identifies observations. Two new variables are created, one to hold the matching score (scaled 0-1) and one for the merge variable. In addition, all of the matching variables from the using dataset are brought into the master dataset (with newly prefixed names) to allow for manual review of matches.

To enhance the speed of this often slow procedure, or-blocking can be used which requires at least one variable to match perfectly between datasets. Or-blocking is the default if 4 or more variables are specified.

Note and Warning

In general, record linkage methods are imperfect and results should be manually reviewed, especially for observations with lower matching scores. It is not uncommon to try several runs with a variety of weights, orblock options, and derived variables to increase the accuracy of the linkage. A series of reclink commands can be used with the help of the exclude option.

Required Options

idmaster(varname) is required and specifies the name of a variable in the master dataset that uniquely identifies the observations. This variable is used to track observations. If a unique identifer does not exist, one can be created simply as gen idmaster=_n.

idusing(varname) is required and specifies the name of a variable in the using dataset that uniquely identifies the observations analogous to idmaster.

gen(newvarname) is required and specifies the name of a new variable created by reclink to store the matching scores (scaled 0-1) for the linked observations.

Common Options

wmatch(numlist) specifies the weights given to matches for each variable in varlist. Each variable requires a weight, although a default of 1 will be used for all variables if not specified. Weights must be >=1 and are typically integers from 1 to 20. The values should reflect the relative likelihood of a variable match indicating a true observation match. For example, a name variable will often have a large weight such as 10 but a city variable, where many duplicates are expected, may have a weight of just 2.

wnomatch(numlist) specifies the weights given to mismatches for each variable in the varlist. These weights are analogous to wmatch weights, but instead reflect the relative likelihood that a mismatch on a variable indicates that the observations don't match -- a small value indicates that mismatches are expected even if the observations truly match. A variable such as telephone number may have a large wmatch but a small wnomatch because matches are unlikely to occur randomly, but mismatches may be fairly common due to changes in phone numbers over time or multiple phone numbers owned by the same person/entity.

orblock(varlist | none) is used to speed up the record linkage by providing a method for selecting only subsets of observations from the using dataset to search for matches. Only observations that match on at least one variable in the Or-Block are examined. Or-blocking on the full varlist is the default behavior if there are 4 or more variables specified. This default can be overriden by specifying orblock(none), which is advised if all of the variables are expected to be fairly unique. New variables are sometimes created in the master and using datasets to assist with Or-Blocking, such as initials of first and last names, street numbers extracted from addresses, and telephone area codes. Or-Blocking can dramatically improve the speed of reclink.

required(varlist) allows the user to specify one or more variables that must match exactly for the observation to be considered a match. The variable(s) must also be in the main varlist and are included in the matching score. This option could have been named andblock to make it's function clear in relation to orblock.

exclude(filename) allows the user to specify the name of a file that contains previously matched observations, providing a convenient way to use reclink repeatedly with different specifications. The exclude file must include the variables specified in idmaster and idusing. Any observation with non-missing values for both id variables is considered matched and is excluded from the datasets for the current matching. Results from each run of reclink can be appended together and specified as the exclude file. This approach can speed up the matching process by starting with more restrictive orblock and/or required specifications that work quickly, followed by a more exhaustive and slow search for the more difficult observations.

Less Commonly Used Options

_merge(varname) specifies the name of the variable that will mark the source of each observation. The default name is _merge(_merge).

exactstr(varlist) allows the user to specify one or more string variables where the bigram string comparator is not used to assess the degree of agreement, but instead the agreement is simply 0 or 1.

uvarlist(varlist) allows the using dataset to have different variable names than the master dataset for the variables to be matched. If specified, the uvarlist must have the same number of variables in the same ordering as the master varlist.

uprefix(string) allows changing the prefix used for renaming the variables in the matching varlist that are brought into the master dataset from the using dataset. The default uprefix is U. For example, if the matching variables are name and address, then the resulting dataset will have variables Uname and Uaddress added from the using dataset for the matching observations.

minscore(#) specifies the minimum overall matching score value (0-1) used to declare two observations a match, default=0.6. Observations in the using dataset are only merged into the master dataset if they have a match score>=minscore and are the highest match score in the using dataset. Lower values of minscore will expand the number of matches but may lead to more false matches.

minbigram(#) specifies the bigram value needed to declare two strings as likely matched, default=0.6. Each raw bigram score is transformed into match and non-match weight multipliers that vary from 0 to 1 with a sharp change at minbigram. A higher value of minbigram may be useful when matching longer strings.

Example

. reclink fname lname address zip phone using bigset, gen(myscore) idm(id) idu(recno) wmatch(3 8 10 2 8) wmnomatch(4 5 8 4 2)

finds matches between current dataset and bigset based on 5 variables. Uses orblocking by default so that only records that match on fname or lname or address or zip or phone will be examined. Could specify orblock(none) to widen possible matches but much slower. Could gen initials=substr(fname,1,1)+substr(lname,1,1) in both datasets and then add initials to the varlist to increase likelihood that or-blocking will work.

Log file segment for a "reclink" job

For the linkage of 2007 survey records to the 2005 baseline table. The function *egen newvar = tag(recordno)* and explicit subscripting are used to list duplicates.

```
. * Linking 2007 data into 2005 list:
.
. use "C:\[path]\KibondoBaseline2005.dta", clear

. count
  571
```

Without weighting of variables

```
. reclink  village namelc using "C:\[path]\KibondoPosition2007.dta", idmaster( recid2005)
idusing( recid2007) gen(matchsc) minscore(0.9)

105 perfect matches found
Added: recid2007= identifier from C:\[path]\KibondoPosition2007.dta    matchsc = matching
score
Observations:  Master N = 571    C:\[path]\KibondoPosition2007.dta N= 893
  Unique Master Cases: matched = 408 (exact = 105), unmatched = 163

.
. save "C:\[path]\Kibondo2005Absorbing2007MinScore90pc.dta"

. * Finding duplicates of the 2005 records in the joined table:

. sort  recid2005

. egen tagrecid2005 = tag( recid2005)

. tab  tagrecid2005
```

```
tag(recid20 |
        05) |      Freq.     Percent        Cum.
------------+-----------------------------------
          0 |          2        0.35        0.35
          1 |        571       99.65      100.00
------------+-----------------------------------
      Total |        573      100.00
```

```
* Using explicit subscripting to list originals as well as duplicates. Unamelc and
Uvillage are the names (lower case) and villages of the "using file", i.e. the 2007
survey table:

. list  recid2005  namelc Unamelc  recid2007   Uvillage matchsc  tagrecid2005 if
tagrecid2005==0 |  tagrecid2005[_n+1]==0, noobs sep(0)

+------------------------------------------------------------------------------------+
| rec~2005           namelc           Unamelc   rec~2007   Uvillage   matchsc   tag~2005 |
|------------------------------------------------------------------------------------|
|      142    julia wilbard   doralia wilbart        789   Kumbanga    0.9423          1 |
|      142    julia wilbard   merania wilbard        788   Kumbanga    0.9423          0 |
|      227   jaclina antony   jekelina antoni        561   Kiduduye    0.9309          1 |
|      227   jaclina antony   katelina antoni        558   Kiduduye    0.9309          0 |
+------------------------------------------------------------------------------------+
```

```
* The variable _merge is added automatically, with 3:"matched", 1:"without match":
. tab  _merge

     _merge |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |        163       28.45       28.45
          3 |        410       71.55      100.00
------------+-----------------------------------
      Total |        573      100.00
```

```
* giving the linkage result for each of the five villages:

. tab  village _merge

           |        _merge
   Village |         1          3 |     Total
-----------+----------------------+----------
  Kiduduye |        15         86 |       101
 Kigendeka |        12         88 |       100
  Kumbanga |       104         67 |       171
 Kumuhasha |        10         90 |       100
  Mukabuye |        22         79 |       101
-----------+----------------------+----------
     Total |       163        410 |       573
```

With weighting

```
. use "C:\[path]\KibondoBaseline2005.dta", clear

. reclink village namelc using C:\[path]\KibondoPosition2007.dta, idmaster( recid2005)
idusing( recid2007) gen(matchsc) minscore(0.9) wmatch(1 5)

105 perfect matches found

Added: recid2007= identifier from C:\[path]\KibondoPosition2007.dta   matchsc = matching
score
Observations:  Master N = 571    C:\[path]\KibondoPosition2007.dta N= 893
  Unique Master Cases: matched = 364 (exact = 105), unmatched = 207

.[renaming of some variables not shown here]


. tab  _mergeUnweighted _mergeWeighted
```

```
_mergeUnwe |    _mergeWeighted
    ighted |         1          3 |     Total
-----------+----------------------+----------
         1 |       163          0 |       163
         3 |        44        364 |       408
-----------+----------------------+----------
     Total |       207        364 |       571

. * Thus 44 fewer matches when these weights were used.
. * Where there is a weighted match, is it the same name as in the unweighted?
.
. gen byte Same2007name = ( UnamelcWeighted== UnamelcUnweighted)

. replace  Same2007name=. if  _mergeUnweighted==1 |  _mergeWeighted==1
(207 real changes made, 207 to missing)

. tab  Same2007name, missing

Same2007nam |
          e |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |        364       63.75       63.75
          . |        207       36.25      100.00
------------+-----------------------------------
      Total |        571      100.00

. * Thus no different matches where both procedure produce some.
```

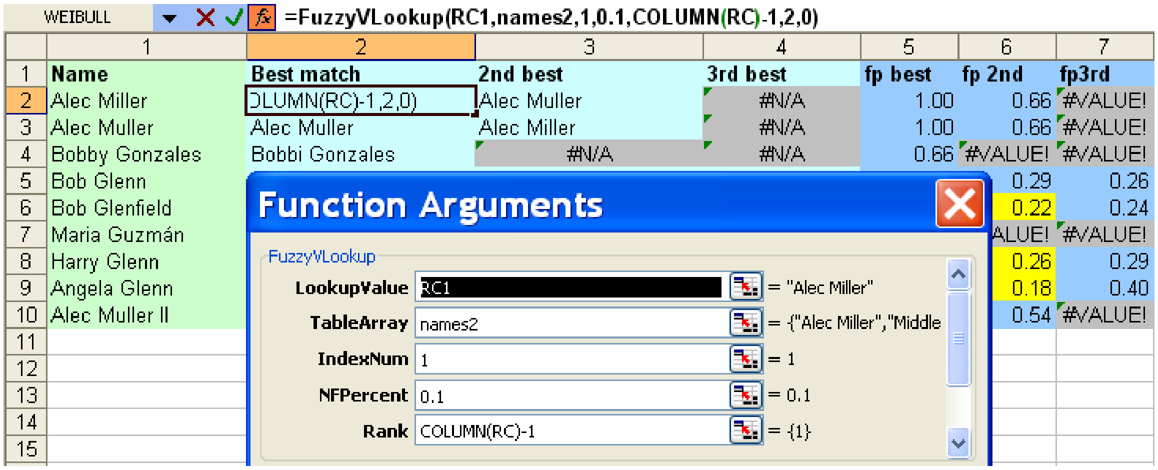
False positives and false negatives

```
. * What kinds of matches did the weighted procedure exclude?
. * Some of the 44 cases no longer matched:

. sort  recid2005

. list  village if  _mergeUnweighted==3 &  _mergeWeighted==1
. * namelc = Name in 2005, UnamelcUnweighted = Name in 2007

     +--------------------------------------------------------+
     |   village                 namelc      UnamelcUnweighted |
     |--------------------------------------------------------|
  6. |  Kumbanga            stazia maro           stazia mwolo | Unweighted false pos.
 28. |  Kumbanga         nashon rukemwa         justin rukemwa | Unweighted false pos.
 50. |  Kumbanga      gabriel sekabanka       milali sekabanka | Unweighted false pos.

[some output omitted]

142. |  Kumbanga          julia wilbard        doralia wilbart | julia w., seen before
178. |  Kiduduye                jenesia          jenesia simon | Prob. weighted false neg.
190. |  Kiduduye     edifilda mbalibali   edefirida mbalimbali | Prob. weighted false neg.
191. |  Kiduduye              elizabeth         elizabet tadeo | More information needed
214. |  Kiduduye         joslin kahozwa        joslini kahodya | More information needed

227. |  Kiduduye         jaclina antony         jekelina antoni | Prob. weighted false neg.
230. |  Kiduduye           yusuph ntulo          yusuphu nturo | Prob. weighted false neg.

[additional output omitted]
```

Inspecting several match candidates with "FuzzyVLookup"

The Excel VBA code for this used-defined function and two associate functions can be downloaded, together with some explanations, from http://www.mrexcel.com/forum/showthread.php?p=955137 (Anonymous 2006)[12].

"FuzzyVLookup" takes seven arguments, of which the last four are optional. The required arguments *LookupValue, TableArray and IndexNum* work similarly to the first three in the deterministic version "vlookup", with some subtle differences in *IndexNum* that need not distract learners. The mechanics of "vlookup" can be studied in the Excel Help function. The image below shows the first part (before scrolling down) of the function arguments wizard which one calls up by clicking on the little box 'fx', to the left of the full function text *"=FuzzyVLookup(RC1,names2,1,0.1,COLUMN(RC)-1,2,0)"*.



WEIBULL =FuzzyVLookup(RC1,names2,1,0.1,COLUMN(RC)-1,2,0)

	1	2	3	4	5	6	7
1	Name	Best match	2nd best	3rd best	fp best	fp 2nd	fp3rd
2	Alec Miller	OLUMN(RC)-1,2,0)	Alec Muller	#N/A	1.00	0.66	#VALUE!
3	Alec Muller	Alec Muller	Alec Miller	#N/A	1.00	0.66	#VALUE!
4	Bobby Gonzales	Bobbi Gonzales	#N/A	#N/A	0.66	#VALUE!	#VALUE!
5	Bob Glenn					0.29	0.26
6	Bob Glenfield					0.22	0.24
7	Maria Guzmán					ALUE!	#VALUE!
8	Harry Glenn					0.26	0.29
9	Angela Glenn					0.18	0.40
10	Alec Muller II					0.54	#VALUE!
11							
12							
13							
14							
15							

Function Arguments

FuzzyVLookup

LookupValue RC1 = "Alec Miller"

TableArray names2 = {"Alec Miller","Middle

IndexNum 1 = 1

NFPercent 0.1 = 0.1

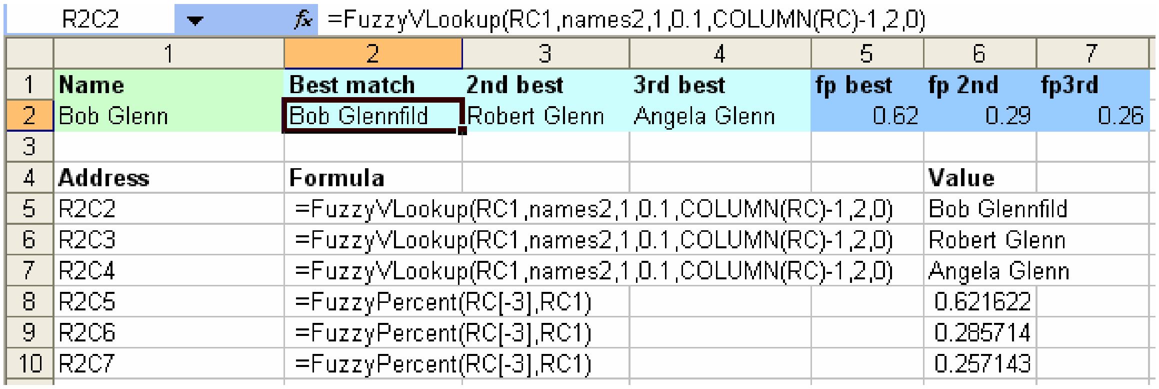
Rank COLUMN(RC)-1 = {1}

Thus a number of basic "vlookup" conventions apply to its fuzzy companion. *LookupValue* defines the variable in the table for which we seek matches from another table, in this example column 1 holding the names. *TableArray* should be a named range, normally the second table. Importantly, the array must hold the corresponding matching variable (in this example again: names) in its leftmost column, just as "vlookup" requires. Practically, this has the consequence that any other formal (e.g. the record number) or substantive variables that are to be imported from the sending table have to be held in the array to the right of the matching variable. *IndexNum* defines the column number in the sending *TableArray* from which the value will be passed. Even though the essence of the procedure is to create matches on the basis of one or several string variables (several if the seventh argument is set > 0), it is good practice to import existing numeric record identifiers as well, into separate columns, of course.

[12] I have not been able to contact the author of this function suite. I therefore refrain from copying the code to this paper. I will be happy to send it, as is, to readers who cannot download it. I assume that users know how to place it in a VBA module (and how to insert a module if there is none attached to the working spreadsheet). If these functions are to be used frequently, it is advisable to place the code in a module attached to the start-up workbook personal.xls. If the saved workbook is to be shared with other users, the functions should be called from a module attached to the workbook itself. For either location, I recommend to insert a new module to hold this code because it includes module-level elements (Option Explicit) that might impair other macros or functions if kept in the same module.

Here, I wish to emphasize the ability of the function to return the best, second-best, etc. match and to propose minor syntactic “tricks” that make it easy to produce a descending sequence of matches horizontally, for quick visual inspection.

In the spreadsheet screenshot below, the three best matches for Bob Glenn are displayed. This example is particularly instructive. Mechanically, the procedure rates “Bob Glennfild” a much better record than the correct “Robert Glenn”. Robert will be selected as the result of visual inspection only.



R2C2 fx =FuzzyVLookup(RC1,names2,1,0.1,COLUMN(RC)-1,2,0)

	1	2	3	4	5	6	7
1	**Name**	**Best match**	**2nd best**	**3rd best**	**fp best**	**fp 2nd**	**fp3rd**
2	Bob Glenn	Bob Glennfild	Robert Glenn	Angela Glenn	0.62	0.29	0.26
3							
4	**Address**	**Formula**				**Value**	
5	R2C2	=FuzzyVLookup(RC1,names2,1,0.1,COLUMN(RC)-1,2,0)				Bob Glennfild	
6	R2C3	=FuzzyVLookup(RC1,names2,1,0.1,COLUMN(RC)-1,2,0)				Robert Glenn	
7	R2C4	=FuzzyVLookup(RC1,names2,1,0.1,COLUMN(RC)-1,2,0)				Angela Glenn	
8	R2C5	=FuzzyPercent(RC[-3],RC1)				0.621622	
9	R2C6	=FuzzyPercent(RC[-3],RC1)				0.285714	
10	R2C7	=FuzzyPercent(RC[-3],RC1)				0.257143	

The lower part of the image opens the formulae used in the six calculated cells of this one sample record. Note that they are so constructed as to be identical for the lookup, respectively for the score calculation. They can simply be dragged to the right hand side in order to produce the second best, third best matches as well as the associated matching scores.

The devices used in this efficient replication are:

- Mixed references
- The function “Column”

The mixed reference RC1 in “FuzzyVLookup” ensures that, wherever the formula is copied in the row, it will collect the first argument from column 1, which is the column holding the names for which matches are sought. “FuzzyPercent” here uses two types of references. The relative reference RC[-3] collects the first argument three columns to the left, which is where a match has been placed, and then as the formula is copied to the right moves with the best, second best, etc. matches. In the second argument, the mixed reference RC1 does the same as in “FuzzyVLookup”; it makes sure the argument is procured from the first column.

“Column(RC)-1” is used for the second of the four optional arguments in “FuzzyVLookup”. It returns the cell’s own column number, then applies a small correction so chosen (-1) that the result is exactly the rank of the match that we want to place in this cell. For example, in column 2, reserved for the best match, the expression supplies *Column(RC)-1 = 2 – 1 = 1*, which is the rank desired here.

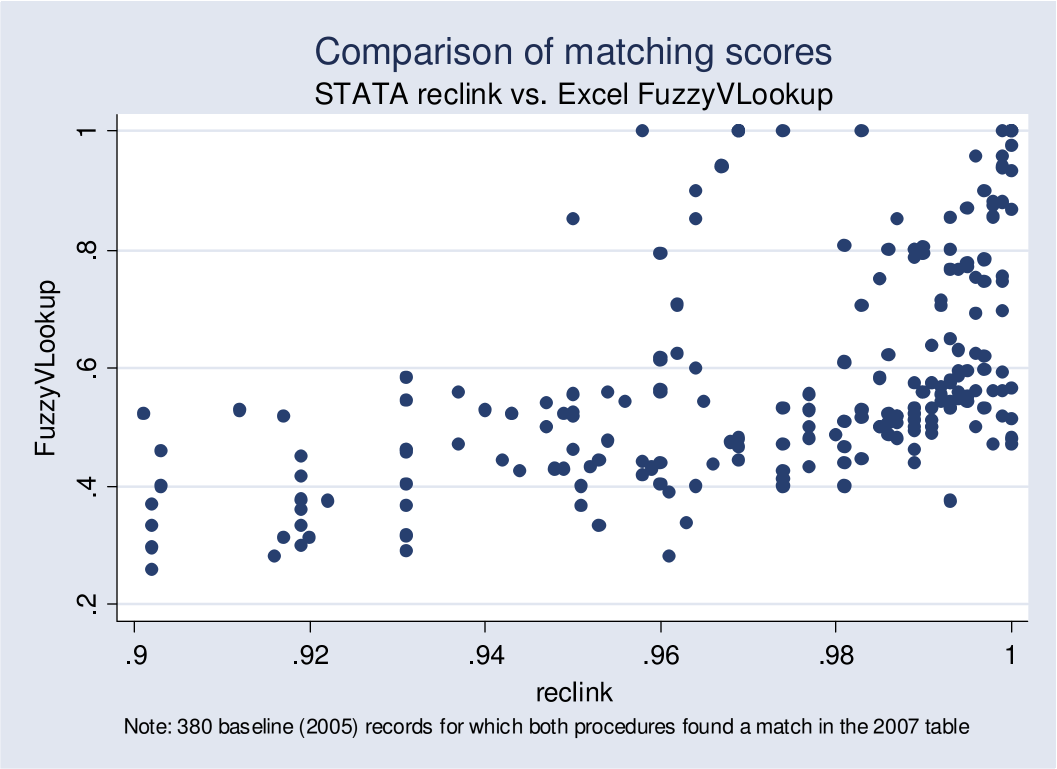
All the seven arguments are described in the code and the explanations on the above-referenced Web site. When the task is not to produce several matches, but rather to import the values in a number of variables for one match (the best!), the "column" function can be exploited, with suitable correction, in the argument *IndexNum*.

As the author warns, "FuzzyVLookup" executes slowly when the tables involved are large. To avoid lengthy freezes during preparation, or while working on other parts of the workbook, user may want to set calculation to "Manual", through the menu commands Tools – Options – Calculate. When it is time to compute the matches, hitting F9 will trigger manual calculation.

Matching scores in "reclink" and "FuzzyVLookup" compared

The way the two applications calculate matching scores are not immediately transparent, and I have not investigated the parts of their codes that achieve this. "reclink" returns scores in the matched table. To visualize scores produced by "FuzzyVLookup", one needs to re-calculate them with its companion "FuzzyPercent".

A comparison of the scores calculated from the Tanzania tables shows that the "FuzzyVLookup" scores disperse over a great range even for matches that both procedures returned identical. This finding is the basis of my recommendation (see page 14) to use "FuzzyVLookup" in a two-step procedure.



Readers anxious to understand the mechanics of matching score calculation completely may want to look into this part of the code in greater depth.

References

Anonymous. (2006). "Fuzzy matching - new version plus explanation [Pseudonym: al_b_cnu]." Manchester, MrExcel.com Retrieved 10 Jun 2008, from http://www.mrexcel.com/forum/showthread.php?p=955137.

Benini, A. (2008). Does Empowerment Work? Underlying concepts and the experience of two community empowerment programs in Cambodia and Tanzania. Washington DC.

Benini, A. and Conley, C. E. (2007). "Rapid Humanitarian Assessments and Rationality: A Value-of-Information Study of a Recent Assessment in Iraq." Disasters 31(1): 29–48.

Benini, A. A. (1991). "Computer Simulation as a Means of Dialogue between Relief Agencies and Local Committees: A Case from Southern Sudan." Disasters 15(4): 331-339.

Blasnik, M. (2001). _GSOUNDEX: Stata module to implement soundex algorithm," Statistical Software Components S420901. Boston, Boston College, Department of Economics.

Blasnik, M. (2007). Record Linkage in Stata [revised 25 Sep 2007]. Boston, North American Stata Users' Group Meetings 2007 5, Stata Users Group.

Blasnik, M. (2008). Personal communication with Aldo Benini [3 June 2008], M. Blasnik & Associates.

Brunborg, H., Lyngstad, T. H. and Urdal, H. (2003). "Accounting for Genocide: How Many Were Killed in Srebrenica?" European Journal of Population 19(3): 229–248.

CDC. (2008). "Link Plus." Atlanta, Division of Cancer Prevention and Control, National Center for Chronic Disease Prevention and Health Promotion Retrieved 3 June 2008, from http://www.cdc.gov/cancer/npcr/tools/registryplus/lp.htm.

Fellegi, I. P. and Sunter, A. B. (1969). "A theory for record linkage." Journal of the American Statistical Association 64(328): 1183-1210.

Herzog, T. N., Scheuren, F. and Winkler, W. E. (2007). Data quality and record linkage techniques. New York; London, Springer.

Machado, C. J. and Hill, K. (2004). "Probabilistic record linkage and an automated procedure to minimize the undecided-matched pair problem." Cadernos de Saúde Pública 20(4): 915-925.

Manning, C. D. and Schütze, H. (1999). Foundations of statistical natural language processing. Cambridge, Mass., MIT Press.

Nkya, E. A. (2008). Impact Data from Kibondo CEP 2007 [E-mail to Duane Poppe, LWF Geneva, 21 January 2008. Appendix 4: Spreadsheet]. Kibondo, Tanganyika Christian Refugee Service.

Nkya, E. A. and Chago, S. (2007). CEP Evaluation 2007.xls [Kibondo marginalized household data, 2 spreadsheets]. Kibondo, Tanganyika Christian Refugee Service.

Schnell, R. and Bachteler, T. (2004). "A Toolbox for Record Linkage." Austrian Journal of Statistics 33 (1&2): 125-133.

Sphere Project (2004). Humanitarian Charter and Minimum Standards in Disaster Response. Geneva, The Sphere Project 2004.

Telford, J. (1997). Good Practice Review 5: Counting and Identification of Beneficiary Populations in Emergency Operations: Registration and its Alternatives. London, Relief and Rehabilitation Network / Overseas Development Institute.

United Nations (2008). Global Symposium +5 on Information for Humanitarian Action, Geneva, 22 - 26 October 2007. Geneva, ReliefWeb and UNOCHA.

UNOCHA (2001). Field Guide for the Use of Geo-Codes. Geneva, United Nations Office for the Coordination of Humanitarian Affairs [In cooperation with and produced by the International Center for Remote Sensing Education].

Urdal, H. (2008). Personal communication to Aldo Benini [6 June 2008]. Oslo, PRIO.

Wikipedia. (2008). "Soundex." Retrieved 16 June 2008, from http://en.wikipedia.org/wiki/Soundex.

About the author

Aldo Benini has a dual career in rural development, with a focus on Bangladesh and another on organizations of the poor, and in humanitarian action. In the latter capacity, he has worked for the International Committee of the Red Cross and for the Global Landmine Survey. He has a Ph.D. in sociology from the University of Bielefeld, Germany, based on field research in community development in West Africa.

Benini is a citizen of Switzerland and an independent researcher based in Washington DC.

He can be contacted at abenini@starpower.net. This and other publications are available at http://aldo-benini.org. The site offers two more technical notes of interest to humanitarian data managers:

"Runs of relief" - A data management technique for humanitarian logistics analysts, and
The Wealth of the Poor. Simplifying living standards measurements with Rasch scales.

27 June 2008