PhUSE 2015

Paper CS06

Detecting Irregularities in Electronical Patient Reported Data

John van Bemmelen, OCS Consulting, 's-Hertogenbosch, The Netherlands

ABSTRACT

In clinical studies the patients may be asked to answer daily a few questions on an electronic device, while not in the clinic. The patients should provide these entries independently, not influenced by clinical personnel or other patients. The date/times of the entries can be used to detect whether data of different patients are entered remarkably simultaneous, indicating potentially fraudulent data entry. Of course such a risk to the validity of a study should be monitored. This paper describes an algorithm to find such combinations of patients, and also gives programming techniques.

INTRODUCTION

In a clinical study the patients were required to enter into an electronic device the data on product intake and on presence of a specific efficacy parameter. This data entry was to occur each morning between 6.00 and 10.00, typically at home. These data were highly important for the study analysis, but no source verification was possible. While glancing through these data some remarkable phenomena were observed: e.g. on one day the entries (earliest to latest) of 7 patients occurred within as little as 3 minutes, while the next day 6 of these patients again had all entries within 3 minutes, but then about an hour later than on preceding day. This seems unlikely if data were really entered independently by the patients.

An algorithm was constructed to detect all pairs of patients whose data entry times over days show remarkable simularities. The algorithm proved very effective in distinguishing between suspicious and nonsuspicious data.

THE SOURCE DATA

The data under investigation[1] consist of one observation for each day on which an entry is expected for each subject. Days on which unexpectedly no entry was made have missing (`.M`) entry time[2].

SUBJID	DATE *(date9.)*	TIME *(time8.)*
1758	12May2014	7:33:46
1758	13May2014	7:42:12
1758	14May2014	M
1758	15May2014	7:39:05
1761	03Jan2014	8:05:01
1761	04Jan2014	7:59:33

A few observations in dataset ENTRIES

ALGORITHM STEP 1: IDENTIFY CLUSTERS OF ENTRIES PER DATE

For each calendar date in the study the entry times of all subjects are inspected. A *cluster* is a set of entries from different subjects with at most 60 seconds between consecutive entry times. For example, the entry times (hh:mm:ss) 6:30:12, 6:30:50 and 6:31:48 are in one cluster. Unexpectedly missing entries for multiple subjects on a date are also considered a cluster. The entries part of a cluster are identified by the cluster number.

While a date may have several clusters, each subject can be part of at most one cluster per date.

Entry time	Subject	
M	2205	*Cluster 1*
M	2249	
6:30:12	2213	*Cluster 2*
6:30:50	2189	
6:31:48	1928	
6:57:10	1755	
7:02:12	2123	*Cluster 3*
7:03:05	2071	
7:03:49	1967	
7:04:24	1834	
7:05:56	1974	

Clusters of entries on a certain date

PROGRAM CODE FOR STEP 1

```
proc sort data=entries (keep=subjid date time);
  by date time;
run;

* Longest seconds-gap between consecutive entries still considered clustering ;
%let timegap=60;                                                    Adjust to your needs

* Intermediate (sequential) cluster numbering: also numbers NON-cluster obs ;
data entries_2 (drop=_prevtime);
  retain _prevtime _clustnr;
  set entries;
  by date time;

  if first.date
    then _clustnr=1;
    else if missing(_prevtime) and not missing(time) then _clustnr+1;
    else if time-_prevtime>&timegap then _clustnr+1;

  _prevtime=time;
run;

* Define definitive cluster number:
  - Ignore non-clusters consisting of only a single observation
  - Number true clusters consecutively, starting at 1 per date
;
data entries_clustnr (keep=subjid date time clustnr);
  set entries_2;
  by date _clustnr;
  if first.date then __clustnr=0;
  if not(first._clustnr and last._clustnr) then
  do;
    if first._clustnr then __clustnr+1;   * also retains __CLUSTNR value ;
    clustnr=__clustnr;
  end;
run;
```

ALGORITHM STEP 2: TRANSFORM SUBJECT-ID AND DATE FOR USE IN ARRAY

For the calculation of scores the data will be searched through in forward and backward direction, for subjects and for dates, whereby the collection of subject-IDs need not be consecutive. The scoring algorithm therefore uses data that are stored in a multi-dimensional array, with subject-IDs and entry dates transformed into consecutive ranges of integers, starting at 1 for respectively the first subject of the study and the first entry date of the study[3].

PROGRAM CODE FOR STEP 2

```
proc sort data=entries_clustnr;
  by subjid;
run;

* determine earliest entry date: day 1 ;
proc sql noprint;
  select min(date) into :day1date
    from entries_clustnr;
quit;

* define sequentially numbered SUBJ to replace SUBJID, and DAY to replace DATE ;
data entries_clustnr_subjday;
  set entries_clustnr;
  by subjid;
  if first.subjid then subj+1;
  day=date-&day1date+1;
```

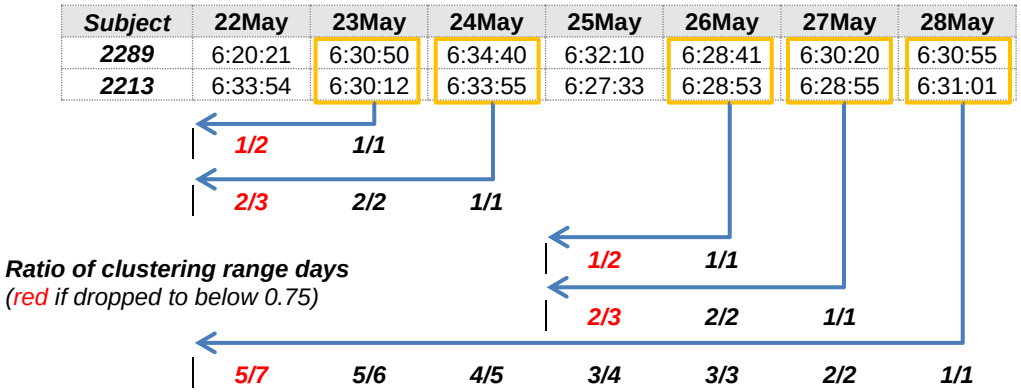
```
run;

* define format SUBJF. to label SUBJ with the value of SUBJID ;
data fmtds (keep=fmtname start label);
  retain fmtname 'SUBJF';
  set entries_clustnr_subjday (rename=(subj=start subjid=label));
  by label;
  if first.label;
run;
proc format cntlin=fmtds;
run;
```

ALGORITHM STEP 3: SCORE THE CLUSTERS PER PAIR OF SUBJECTS

For each pair of subjects in the study the entry times are inspected for all days on which both subjects are expected to have an entry. Each day for which the entries are regarded as part of the *same* cluster will count towards an overall cluster score, which is a measure for the suspiciousness of the entry time pattern. The score increases if the pair of subjects has a range of consecutive days with a cluster. If there are large differences in the entry times of such consecutive days then the score increases even more. For consecutive days with clustered missing entries the score is based on the rareness of missing entries (in the range of days in common between the pair).

AMPLIFYING SCORE FOR PRECEDING CLUSTER DAYS

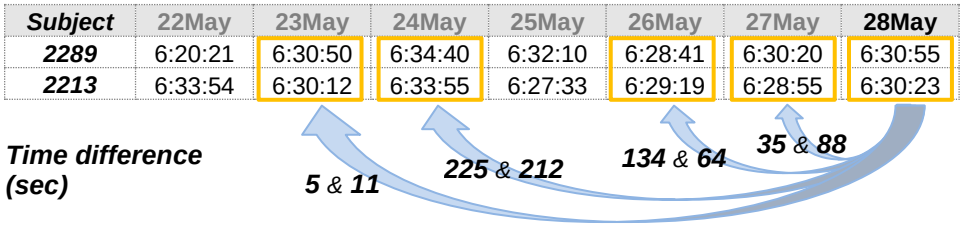
Each day with an entry cluster is compared to all cluster days in a preceding range of days, where the length of this range depends on the ratio of cluster days in the range over the total number of days in the range; an extra day is added to the begin of the range as long as this ratio is at least 0.75.



Example: Ratio of clustering range days and longest range of days back-inspected for several dates

Each day in the range, if clustering, has its entry times compared against those on last day of the range (also a cluster day).
In the above example, 26 May is not compared to any preceding day, 27 May is compared to 26 May, and 28 May to respectively 27 May, 26 May, 24 May and 23 May.

For 28 May the following (absolute) time differences are used in the algorithm:



Example: Time differences for one date

MISSING TIME DIFFERENCES

If the time difference is unknown, due to missing entry time at either side of the compared days, then a formula is used based on the ratio of the number of days in common over the number of days with missing entries for each of the subjects on the days in common.

Subject	9Jul	10Jul	11Jul	12Jul	13Jul	14Jul	15Jul
1856	8:45:54	.	8:55:23	9:10:58	.		
2353			8:10:32	.	.	.	7:34:22

Example: Only 1 and 2 missing entries for these respective subjects, over 3 days in common

For subject 1856 the last expected entry is on 13Jul. For subject 2353 the first expected entry is on 11Jul. So, this subject pair has only 3 days in common, of which 1 is a cluster day. Subject 1856 has 1 missing entry and subject 2353 has 2 missing entries over these 3 days.

The exact formulas for calculating scores are given in the program code below. These formulas have proven themselves, but are arbitrary.

WEIGHING THE SCORE OF A SUBJECT PAIR FOR THE NUMBER OF DAYS IN COMMON

With a higher number of days in common between two subjects the chance of clustered days increases. To adapt for this the sumscore is divided by the number of days in common.

PROGRAM CODE FOR STEP 3

```
* determine highest SUBJ and DAY value to define dimensions of array ;
proc sql noprint;
  select max(subj), max(day) into :maxsubj, :maxday
    from entries_clustnr_subjday;
quit;

* the minimum required ratio of clustered days in a range of consecutive days at
  which it is allowed to continue by adding a preceding day to the range
;
%let minclustratio=0.75;

* Read data of all observations into an array, and then
  For each pair of subjects calculate cluster-score
;
data clusterscores (drop=subj day dy time clustnr currclustrang);
  set entries_clustnr_subjday end=eof;

  * T{i,j} = time    for subj=i on day=j
    C{i,j} = clustnr for subj=i on day=j
  ;
  array t{&maxsubj,&maxday} _temporary_;
  array c{&maxsubj,&maxday} _temporary_;

  t{subj,day}=time;
  c{subj,day}=clustnr;

  * EOF indicates last read observation, i.e. all data have been read into array ;
  if eof then
  do;
    * systematically go through all possible subject-pairs SUBJ1 x SUBJ2;
    do subj1=1 to &maxsubj-1;
      do subj2=subj1+1 to &maxsubj;

        * determine descriptives for the subject-pair:
          earliest/latest/number of days in common (=entry expected for both) /
          number of common days with missing entry for first/second subject
        ;
        * initialise ( Min-Max from 0 to -1, to ensure no days are in-between ) ;
        pairminday=0; pairmaxday=-1; pairdays=0;
        pairdaysmis1=0; pairdaysmis2=0;

        do dy=1 to &maxday;
          if  t{subj1,dy}>=.M  and  t{subj2,dy}>=.M  then
```

Adjust to your needs

```
  do;
    if pairminday<1 then pairminday=dy;
    pairmaxday=dy;
    pairdays=pairdays+1;
    pairdaysmis1=pairdaysmis1+ missing(t{subj1,dy});
    pairdaysmis2=pairdaysmis2+ missing(t{subj2,dy});
  end;
end;

* initialise, where latter 3 are for descriptive purposes only ;
sumscore=0;
daysclustering=0; currclustrang=0; maxclustrang=0;

* evaluate the respective days that subject-pair has in common ;
do dy=pairminday to pairmaxday;

  if c{subj1,dy}^=. and c{subj1,dy}=c{subj2,dy} then
  do;

    daysclustering+1;
    currclustrang+1;
    maxclustrang=max(maxclustrang,currclustrang);

    * score for clustering on day DY itself ;
    sumscore + 1;

    * amplify score for clustered preceding days (DY2):
      evaluate days DY2 preceding DY, as long as range of days has
      required minimum ratio of cluster-days, and DY2 is clustered
    ;
    rngdays=1; rngdaysclust=1;   * initialise range details (for DY only) ;

    do dy2=dy-1 to pairminday by -1;

      if c{subj1,dy2}^=. and c{subj1,dy2}=c{subj2,dy2} then
      do;
        * compare times of first and last day of cluster-range ;
        if missing(t{subj1,dy}) or missing(t{subj1,dy2})
            then sumscore+ 2 * ( pairdays/(pairdaysmis1+pairdaysmis2) )**2;
            else sumscore+ ( ( t{subj1,dy} - t{subj1,dy2} )/60 )**2
                         + ( ( t{subj2,dy} - t{subj2,dy2} )/60 )**2;

        * update range details for added day DY2 ;
        rngdaysclust+1; rngdays+1;
      end;
      else
        rngdays+1;    * update range details for added day DY2 ;

      * if ratio drops below threshold: stop loop to add preceding days ;
      if rngdaysclust/rngdays<&minclustratio then dy2=0;

    end;  * DY2 ;

  end;
  else currclustrang=0;

end;  * DY ;

* weigh the score for number of days in common ;
if sumscore>0 then score=sumscore/pairdays;
```

Formula for day itself (arbitrary)

Formulas for preceding days (arbitrary)

```
                    else score=0;
        * output observation with SUBJ1 x SUBJ2 scoring details ;
        output;

      end;  * SUBJ2 ;
    end;  * SUBJ1 ;
  end;  * EOF ;

  * ensure SUBJID values are shown in every presentation of these data ;
  format subj1 subj2 subjf.;

  label subj1         = '1st subject in the pair'
        subj2         = '2nd subject in the pair'
        pairdays      = 'Days in common between the pair'
        pairdaysmis1  = 'Days in common with missing entry, 1st subject'
        pairdaysmis2  = 'Days in common with missing entry, 2nd subject'
        daysclustering= 'Days with clustered entry between pair'
        maxclustrang  = 'Longest range of clustered entry days'
        sumscore      = 'Summed score over all days'
        score         = 'Score: Sumscore / Pairdays'
  ;
run;
```

RESULTING DATASETS

Two important datasets result from the algorithm: one with all entries in which cluster numbers are identified, and one with the scoring data for all possible pairs of subjects in the study.

SUBJID	DATE *(date9.)*	SUBJ	DAY	TIME *(time8.)*	CLUSTNR
1727	25JUN2014	2	157	7:58:39	6
1727	26JUN2014	2	158	7:46:18	3
1727	27JUN2014	2	159	7:57:14	3
1728	09FEB2014	3	21	M	
1728	10FEB2014	3	22	9:31:14	
1728	11FEB2014	3	23	M	1
1728	12FEB2014	3	24	9:40:37	
1728	13FEB2014	3	25	9:18:45	2

A few observations in dataset ENTRIES_CLUSTNR_SUBJDAY

SUBJ1 *(subjf.)*	SUBJ2 *(subjf.)*	SCORE	SUM-SCORE	PAIR-DAYS	DAYS-CLUSTE-RING	MAX-CLUST-RANG	PAIR-DAYS-MIS1	PAIR-DAYS-MIS2
1724	1728	0.008	1	122	1	1	10	12
1724	1739	0.016	2	126	2	1	11	11
1724	1746	0.011	1	90	1	1	6	4
1724	1759	0.025	2	79	2	1	3	33
1727	1728	96.420	13402.4	139	11	7	1	12
1727	1739	92.615	13243.9	143	11	7	1	11

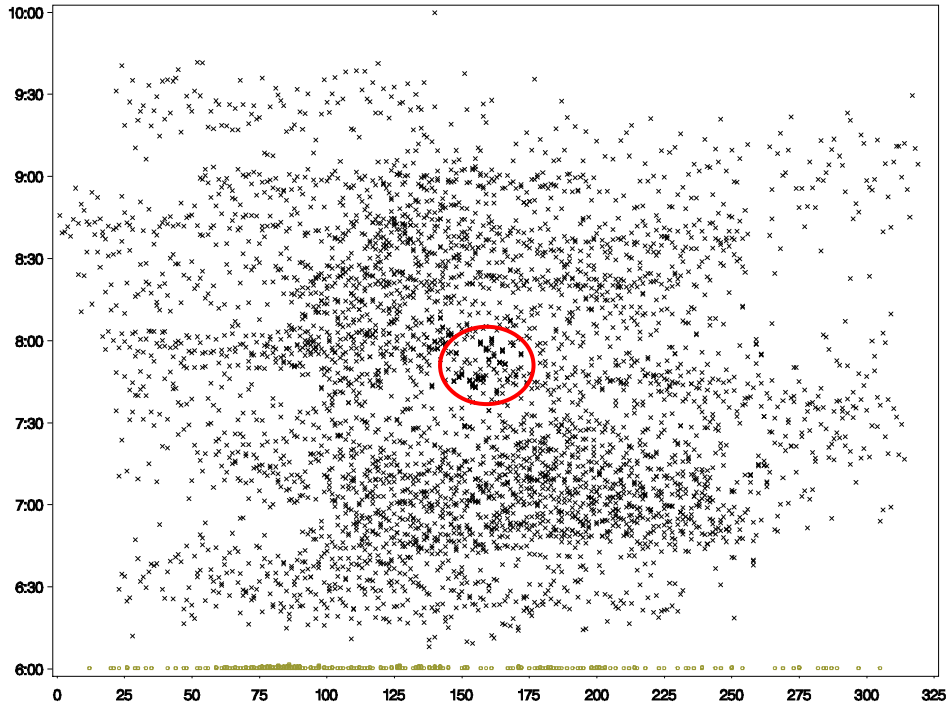
A few observations in dataset CLUSTERSCORES

PRESENTING RESULTS

With these datasets it is quite simple to create informative figures and listings to further investigate the data and specifically those of the identified suspicious subject pairs.

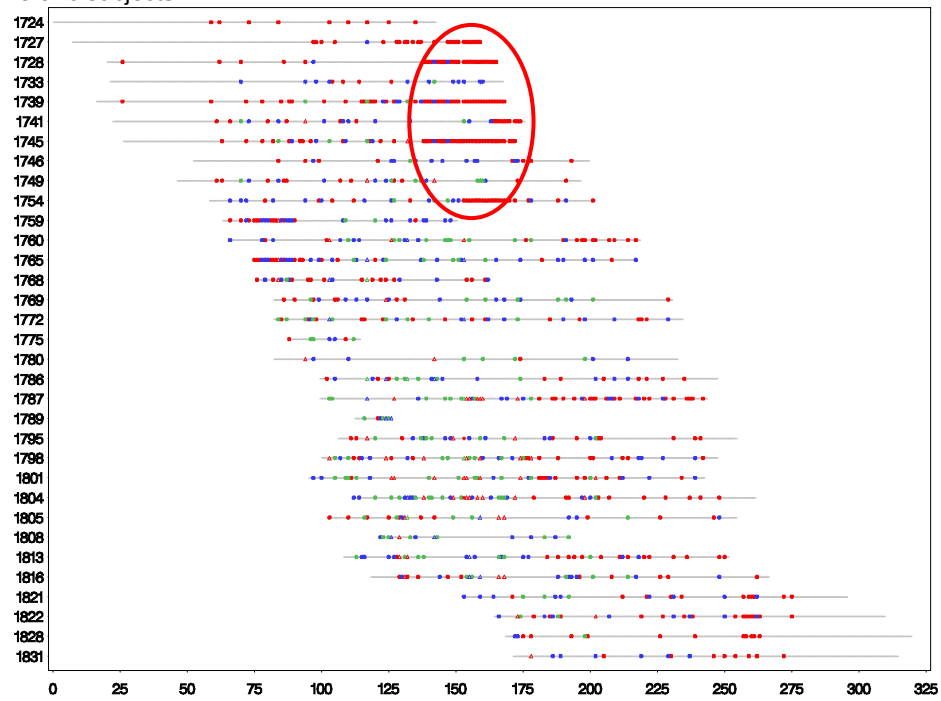
SCATTER PLOT OF ALL ENTRY TIMES OVER ALL DAYS

This figure presents all entries by DAY (horizontal) and TIME (vertical). Each black cross represents an entry, and each green circle (around 6AM) represents a missing entry. Clusters that contain many subjects on a day are visible as clustered symbols, e.g. around day 150 near 8 AM.



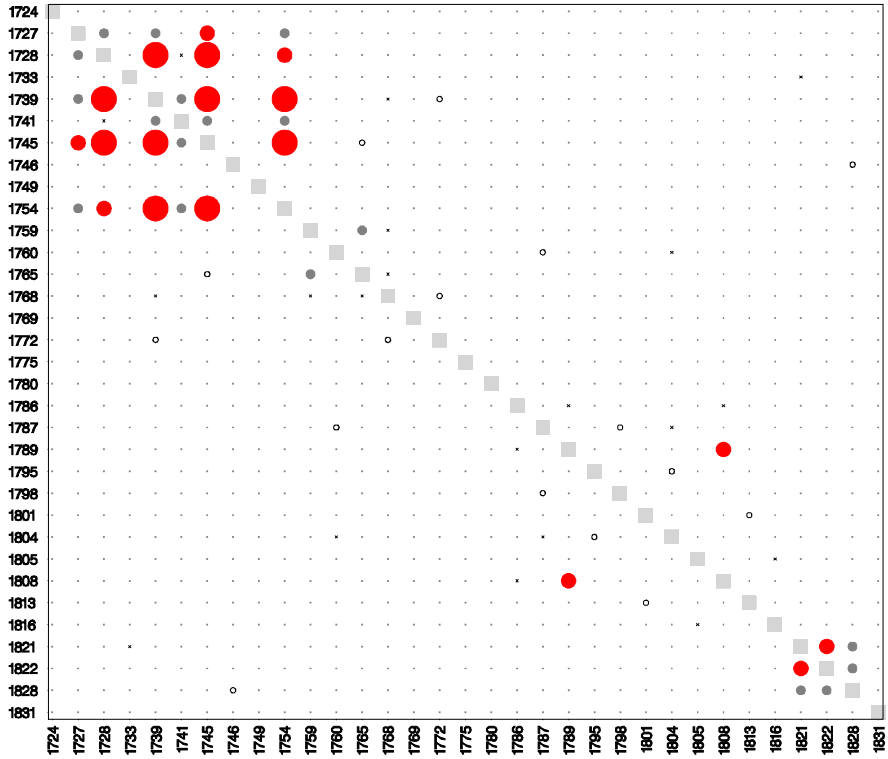
PLOT OF ENTRY CLUSTERS FOR ALL SUBJECTS OVER ALL DAYS

This figure presents all CLUSTNR (coloured symbols) by SUBJ (vertical) and DAY (horizontal). The days for which an entry is expected for a subject are shown as a grey line. The clustering around day 150, as seen in preceding figure, originates from 5 or 6 subjects.



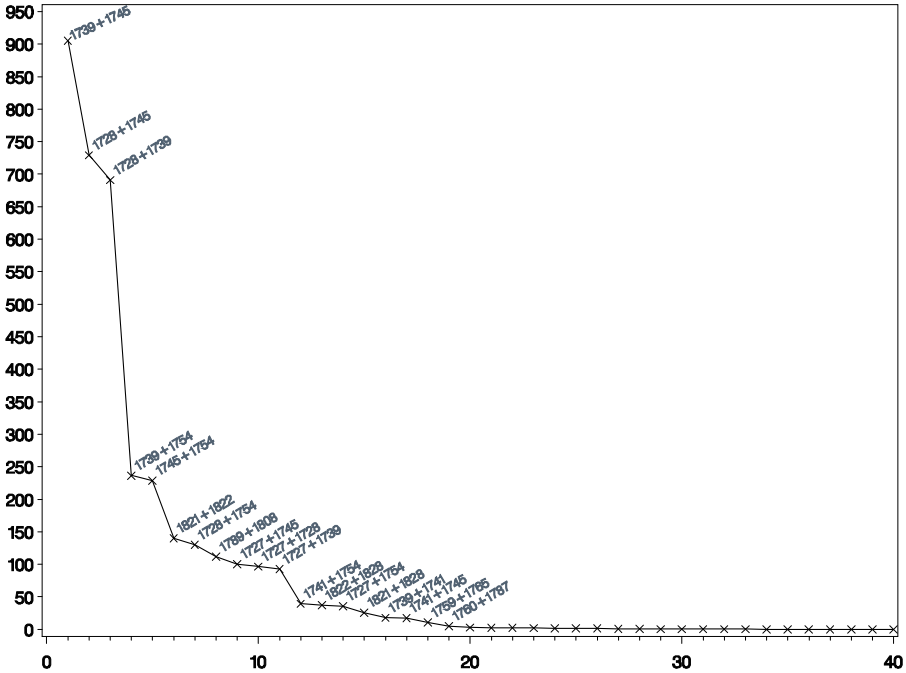
PLOT OF SCORES BETWEEN PAIRS OF SUBJECTS

This figure presents all SCORE values (ranges of values indicated by different symbols) by SUBJ1 and SUBJ2 (on the axes). It is clear that high cluster scores occur for 5 pairs of subjects, consisting of 4 subjects in total: 1728, 1739, 1745 and 1754.



PLOT OF DESCENDING SCORES

This figure presents the SCORE values (vertical) for the first 40 subjects pairs (horizontal), as sorted by descending score. Note that, with 33 subjects included in the algorithm, there are in total 528 possible subject pairs. The subject pairs (SUBJ1 - SUBJ2) for the highest scores are annotated in this figure. It is clear that the scores drop quite quickly, making them a strong discriminating tool.



LISTINGS OF DATA FOR CLUSTERED PAIRS OF SUBJECTS

A high score is a strong indicator for the suspiciousness of the data, but this should also be clear from the individual entry data themselves, which will also be understood better by others. Manual review of the data for the subspected subject pairs (and possibly: subject groups) to substantiate the findings.

Here are a few examples:

Score: 236.49 110 days in common 17 cluster days, max. range: 16 days		
day	**1739**	**1754**
133	M	M
153	7:45:12	7:45:00
154	7:43:54	7:42:56
155	7:42:50	7:42:47
156	7:45:54	7:45:50
157	7:59:36	7:59:06
158	7:46:31	7:47:06
159	7:56:29	7:56:51
160	7:52:52	7:53:04
161	8:00:15	8:00:09
162	7:54:26	7:54:19
163	7:41:50	7:42:03
164	7:51:56	7:51:51
165	7:56:21	7:56:35
166	7:51:31	7:51:31
167	7:46:11	7:44:55
168	8:07:28	8:07:10

The subjects 1739 and 1754 have a long range of consecutively clustering days, but with relatively small time differences between days. Inspection from day 168 backwards contributes more than a third of the total score, because all days from day 153 onwards contribute as preceding day, and the time on day 168 is (much) later than that on any of these preceding days.
This continued clustering combined with jumps in time between days indeed make this a strongly suspicious case.

Score: 111.69 5 days in common 4 cluster days, max. range: 2 days		
day	**1789**	**1808**
122	8:35:00	8:34:50
123	8:31:31	8:31:32
125	8:44:21	8:44:42
126	9:00:49	9:00:50

The subjects 1789 and 1808 have 4 clustering days with only 5 days in common. In the two 2-day cluster ranges the time difference between day 125 and 126 is most suspicious.

Score: 25.81 127 days in common 3 cluster days, max. range: 2 days		
day	**1821**	**1828**
257	7:11:03	7:10:47
260	7:14:40	7:14:15
261	7:55:02	7:54:49

The subjects 1821 and 1828 have only 3 clustered days. The time difference between consecutive cluster days 260 and 261 is large, which makes this a suspicious case.

Score: 3.01 143 days in common 12 cluster days, max. range: 2 days		
day	**1795**	**1804**
120	6:53:13	6:51:41
134	6:44:14	6:44:46
141	6:47:45	6:46:50
148	6:44:01	6:43:18
149	6:51:23	6:52:13
153	6:51:05	6:50:43
168	6:53:10	6:53:23
172	6:53:15	6:52:38
200	6:53:03	6:53:39
202	6:48:27	6:48:40
203	M	M
241	6:45:36	6:45:50

The subjects 1795 and 1804 have only one range of consecutively clustered days, consisting of only 2 days. Time differences between clustered days are very small anyway.
Each of these subjects may simple have a routine time at which they make an entry each day, resulting in clusters only by chance.
In this case an inspection of the entry times on surrounding, non-clustered days may influence the final conclusion.

ADAPTATIONS

FINE TUNING

If the entry device has an alarm that can be set to remind the subject, then entries are more likely in the few minutes following such an alarm window. Such entries should be weighed down in the scoring algorithm.

If besides a daily entry in the morning also an other daily entry, say, in the evening, is requested from the subjects, then the algorithm can also be applied to those data, and scores of the two can be aggregated to enhance the search.

With some adaptations the algorithm can also be used if, say, only one week per 4 weeks have entries required. Per subject, the three weeks in between entry weeks are then considered days without an expected entry.

ALTERNATIVES

The entry times refer to the start of each entry, but if also the stop time of each entry is known then entry durations can also be analysed. In suspicious cases, where one person makes many consecutive entries, entry durations will be much shorter.

Subjects used a dedicated device for their entries, but if entries are made in an app on a smartphone or on a website, then a.o. location information may be captured with each entry, which would certainly simplify fraud investigations. However, it may be ethically or legally prohibited to collect such information during the conduct of a study.

CONCLUSION

The exact formulas and thresholds may be arbitrary, but the algorithm certainly is capable of distinguishing between suspicious and non-suspicious data. Review of the data for these (few) suspected cases is essential.

The clocktime information of entries typically does not play a large role in the formal analyses for study reporting. The algorithm will therefore not jeopardise the power of the study (unless, of course, suspicious data have to be omitted from the study analyses).

Be prepared that extra work will arise if truly suspicious data are detected: the cases need to be explained to a.o. auditors, delicate, legal battles will have to be fought, and the protocol may have to be amended to cope with changed or added analyses.

Nonetheless, patient reported data should be just that! The algorithm presented in this paper can be used to detect violations of this principle, and should be applied wherever applicable.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

John van Bemmelen
OCS Consulting
Ruwekampweg 2G
5222 AT 's-Hertogenbosch
The Netherlands
Work Phone: +31 73 523 6000
Email: john.vanbemmelen(at)ocs-consulting.com
Web: http://ocs-consulting.nl/

[1] In this paper simulated data are used. The original study data contained 1800 subjects, with the algorithm applied for each of the sites separately.

[2] The use of special missing `.M` as opposed to normal missing `.` is essential for the correct functioning of the algorithm: within the array, the former represents a day on which unexpectedly there is no entry, and the latter a day for which no entry is expected for a subject. *Note that in SAS®* `missing(.)` *and* `missing(.M)` *both are true, while* `.<.M.`

[3] Interaction between subjects from different countries can be ruled out by definition, so the scoring algorithm might be applied for each of the countries in the study separately, which improves efficiency.