Macro-selection and micro-editing: a prototype

Wim Hacking, Roger Lemmens
Statistics Netherlands
Problems with data-editing
(read: challenges)

- Less funds supplied
- More electronic input
- More output demanded
More efficient data editing

1. Electronic questionnaires with checks / feedback
2. Automatic corrections / imputations
3. Meso analysis for
   - manual edits only when having effect at the aggregate level
   - Verifying results step 1
   → MacroView
Selection of the microdata

Visualize

Edit

Select subset
MacroView POC

- Text-based (for now)
- Must be flexible
  - For prototyping
  - Final version must be usable for different statistics
- Input from & application to:
  - Transport
  - Production
mAcrodata (aggregated) → mMicrodata → Combined aggregate + plausibility functions
Specification of microdata

DATAMODEL MyData

primary

   ID

weight

   TheWeight

FIELDS

   ID :integer
   InterviewId :integer
   TransportedWeight :real, ifmissing(0.0)
   TransportType :integer, ifmissing(0)
   Distance :real
   TheWeight :real
   Quarter :integer
   NstrCode :integer

RULES

   IF Type = 'X' THEN
       Distance < 2000 "Distance for type X should not exceed 20"
   ENDIF

ENDMODEL
DATAMODEL MyAggregateData

AGGREGATEBY
   AggDef = Quarter * NstrCode

FILTERBY
   Filter1 = ‘Profit > 1000’

FIELDS

   Quarter: integer
   NSTRCode: integer
   Average_Distance :real
   Sum_Weights :real

...
DATAMODEL MyAggregateData

AGGREGATEBY
AggDef = Quarter * NstrCode
AggDef2 = Quarter

FILTERBY
Filter1 = ‘Profit > 1000’
Filter2 = ‘Profit <= 1000’

FIELDS
Quarter: integer
NSTRCode: integer
Average_Distance : real
Sum_Weights : real
...

ENDMODEL
Specification of aggregate calculation: how to detect anomalies?

1. Distribution properties of the micro data, e.g. its variance.

2. Processing properties
   - % item non-response
   - % previously automatically imputed values in field X

3. Plausibility functions: e.g. the relative change between weighted t-1 and t data:

   \[ \Delta = \frac{\left| \sum w_{i,t-1}V_{i,t-1} - \sum w_{i,t}V_{i,t} \right|}{w_{i,t-1}V_{i,t-1}} \]
Comparing aggregate values

<table>
<thead>
<tr>
<th>Quarter</th>
<th>Average_Distance(t-1)</th>
<th>Average_Distance(t)</th>
<th>Delta</th>
<th>CV_Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2000</td>
<td>3500</td>
<td>0.75</td>
<td>4.5</td>
</tr>
<tr>
<td>2</td>
<td>1500</td>
<td>1700</td>
<td>0.14</td>
<td>2.4</td>
</tr>
<tr>
<td>3</td>
<td>2300</td>
<td>2100</td>
<td>0.09 (2,3)</td>
<td>1.6 (1)</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Specification of an aggregate

Aggregate MyAgg

INPUT

CY = MyData
LY_Agg = MyAggregateData

output

outputagg = MyAggregatedDataCompared

cells

Difference := abs(LY_Agg.average_distance - AVG(CY.Distance))/AVG(LY.Distance) ;

Med_Dist := Median( CY.Distance );

Cellcompare

Aggdef2:

ratio := average_distance(1) / average_distance(2)
> 2.0 "There should be more transport in the first quarter!";

EndAggregate

<table>
<thead>
<tr>
<th>Quarter</th>
<th>Average_Distance</th>
<th>Sum_Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.91</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>0.89</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Specification of a grid

Grid Kerncel_Grid
input = MyAggregateData

rules

color(average_distance, difference, 0, 30.0, Green, Red)

EndGrid
Specification of a plot

Plot MicroPlot
input = MyData

type = scatter

xcolumn = Distance
YColumn = TransportedWeight
ZColumn = TheWeight
mincolorvalue = 1.0
maxcolorvalue = 500.0
color = (Red, Blue, Green)

TooltipText='Weight=\$THEWEIGHT\$
endplot
Specification of the proces

Proces

Agg_LY(Data_LY, DataMacro_LY)
Agg_LY.Done -> MyAgg(Data, DataMacro_LY, DataMacro)
MyAgg.done -> comparegrid1(av.outputdata) AT Demo Position Top
Label 'DataCompared'

comparegrid1.RecordsSelected ->
MicroPlot(comparegrid1.SelectedRecords) AT Demo POSITION Tab
Label 'SelectedRecords'

comparegrid1.RecordsSelected ->
Details2(comparegrid1.SelectedRecords) AT Demo POSITION Left
Label 'MicroData'

MicroPlot.RecordsSelected -> MyEdit(MicroPlot.SelectedRecords)
MyEdit.RecordEdited -> MyAgg(Data, DataMacro_LY, DataMacro)

Endproces
Specification of the process (2)
Micro-editing
MacroView: currently

- Testing with real data
  - Road transport data (~demo)
  - Production statistics
  - Generating new ideas for the POC

- Investigating the possible integration with Blaise
  - Data layer
  - Syntax

- Making the POC more robust
  - Better checking of illegal syntax
A demo...
Questions ?
Grid met aggregaat + aankleuring

<table>
<thead>
<tr>
<th>ID</th>
<th>KERNCEL</th>
<th>btw_totaal</th>
<th>omzet_gemiddeld</th>
<th>plausibiliteitsindex</th>
</tr>
</thead>
<tbody>
<tr>
<td>12806</td>
<td>29100</td>
<td>59530</td>
<td>1175,92</td>
<td>1</td>
</tr>
<tr>
<td>12844</td>
<td>29300</td>
<td>65214</td>
<td>1758,9211</td>
<td>1</td>
</tr>
<tr>
<td>12852</td>
<td>29400</td>
<td>9206</td>
<td>1118,625</td>
<td>1</td>
</tr>
<tr>
<td>12883</td>
<td>29500</td>
<td>46303</td>
<td>1346,4839</td>
<td>1</td>
</tr>
<tr>
<td>12884</td>
<td>29600</td>
<td>364</td>
<td>361</td>
<td>0</td>
</tr>
<tr>
<td>12892</td>
<td>29700</td>
<td>11663</td>
<td>1029,875</td>
<td>1</td>
</tr>
<tr>
<td>12916</td>
<td>31100&lt;</td>
<td>43278</td>
<td>1795,1567</td>
<td>1</td>
</tr>
<tr>
<td>12948</td>
<td>31620&lt;</td>
<td>62377</td>
<td>1874,5625</td>
<td>1</td>
</tr>
<tr>
<td>12967</td>
<td>32100</td>
<td>54011</td>
<td>2176,8421</td>
<td>1</td>
</tr>
<tr>
<td>12973</td>
<td>32300&lt;</td>
<td>6401</td>
<td>1142</td>
<td>1</td>
</tr>
<tr>
<td>12975</td>
<td>33101</td>
<td>531</td>
<td>259,5</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i/deld</th>
<th>PLAUSIBILITEITSI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>115/8</th>
<th>1512U</th>
<th>/b/</th>
<th>963,125</th>
</tr>
</thead>
<tbody>
<tr>
<td>11592</td>
<td>15130</td>
<td>48936</td>
<td>3430,2857</td>
</tr>
<tr>
<td>11605</td>
<td>15200</td>
<td>28904</td>
<td>2190</td>
</tr>
<tr>
<td>11611</td>
<td>15310</td>
<td>7565</td>
<td>1224,6667</td>
</tr>
<tr>
<td>11617</td>
<td>15330&lt;</td>
<td>14821</td>
<td>2512,1667</td>
</tr>
<tr>
<td>11621</td>
<td>15400</td>
<td>16733</td>
<td>4129,25</td>
</tr>
<tr>
<td>11654</td>
<td>15500</td>
<td>16498</td>
<td>445,7879</td>
</tr>
<tr>
<td>11667</td>
<td>15610</td>
<td>16612</td>
<td>1163,4615</td>
</tr>
</tbody>
</table>
Plots met aankleuring/zoom/selectie
Todo/vragen:

Met echte data aan de slag bij V&V en Mesogaafmaken

Vervolg: waar moet de definitieve versie gemaakt worden? Binnen een project of bij Blaise?
Select data source(s)

- Select of data sources occurs through so called .boi file(s)
- Presently only one data source can be selected
- Version for 2001: multiple data sources
  - record set data
  - aggregated data: table meta (Cristal)
Define groups

- Grouping variable types are:
  - enumeration
  - integer/ float
  - classification

- Group the data, e.g.
  - sex (male, female)
  - turnover (0..100, 101..200, 201..300)

leads to 6 groups
Define/select variables to observe

- select variables, e.g.
  - TotalSalaries, NoPersons

- Define derived variables, if needed, e.g.
  - AvgSalary := TotalSalaries / NoPersons

- Generate table grid

- Choose indicators to analyse:
  - Mean, median, MAD, min, max, variance, standard deviation, alpha-trimmed mean, cell-filling, number of records.
Show aggregates in grid:

<table>
<thead>
<tr>
<th>Number</th>
<th>Kop_Gk</th>
<th>Kop_Telnr</th>
<th>l_Werkn090_Al_InUim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>2640</td>
<td>2640</td>
<td>2640</td>
</tr>
<tr>
<td>Group 2</td>
<td>930</td>
<td>930</td>
<td>930</td>
</tr>
<tr>
<td>Group 3</td>
<td>843</td>
<td>843</td>
<td>843</td>
</tr>
<tr>
<td>Group 4</td>
<td>490</td>
<td>490</td>
<td>490</td>
</tr>
<tr>
<td>Group 5</td>
<td>250</td>
<td>250</td>
<td>250</td>
</tr>
<tr>
<td>Group 6</td>
<td>352</td>
<td>352</td>
<td>352</td>
</tr>
<tr>
<td>Group 7</td>
<td>786</td>
<td>786</td>
<td>786</td>
</tr>
</tbody>
</table>
Define/select variables to observe

Alternative aggregate functions

CV(a) = std(a) / mean(a)

<table>
<thead>
<tr>
<th></th>
<th>Std</th>
<th>Mean(a)</th>
<th>a</th>
<th>CV(a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>100.2</td>
<td>3.4</td>
<td>0.034</td>
<td></td>
</tr>
<tr>
<td>Group 2</td>
<td>90.4</td>
<td>6.2</td>
<td>0.069</td>
<td></td>
</tr>
</tbody>
</table>

Define checks on data (later: using checks from Blaise)

- abs(profit97 - profit98)*2/(profit97 + profit98) < 0.1 --> percentage
- Filling(NoPersons) < 0.3 (non-response) --> signalling flag

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>profit97</th>
<th>profit98</th>
<th>Δ profit</th>
<th>Filling &lt; 0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>234</td>
<td>250</td>
<td></td>
<td>2.3%</td>
<td></td>
</tr>
<tr>
<td>Group 2</td>
<td>456</td>
<td>503</td>
<td></td>
<td>14.7%</td>
<td></td>
</tr>
</tbody>
</table>
Macro level: Detection of suspicious cells

Identify suspicious data in cells (outliers), which have a potential impact on survey estimates using:

- Estimates of the distribution (mean, std, …)
- Manual outlier detection using boolean expression variables
- Automatic outlier detection using Kosinsky algorithm
Outlier detection: Kosinsky

- 1-D: use $|x - \text{median}(x)|/\text{MAD}(x)$ distance to robustly detect outliers
- N-D: No median defined, use Mahalanobis distance instead:
  \[
  M^2 = \Sigma (y_i - \mu)^T C^{-1} (y_i - \mu) \quad ( (y_i - \mu)/\sigma \text{ in 1D})
  \]
Kosinski-algorithm

1. Start with $n_0 = 0.1n$ ‘good’ points
2. Good points $\Rightarrow \mu$
3. Obtain all Mahalanobis distances
4. Take the $(1+f)n_i$ points with the smallest distances if distance $< \text{cutoff}$
5. $n_{i+1} = (1+f)n_i$
6. Repeat until no more points added.
Parameters for outlier detection: f and cutoff
Grid after outlier detection
Meso level: Detection of points in plots

Identify suspicious data in XY plots (outliers), which have a potential impact on survey estimates using:

- Manual outlier detection using boolean expression variables
- Automatic outlier detection using Kosinsky
Scatter plot with outliers marked blue:
Define/select variables to observe

Alternative aggregate functions

\[ CV(a) = \frac{\text{std}(a)}{\text{mean}(a)} \]

<table>
<thead>
<tr>
<th>Std</th>
<th>Mean(a)</th>
<th>a</th>
<th>CV(a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>100.2</td>
<td>3.4</td>
<td>0.034</td>
</tr>
<tr>
<td>Group 2</td>
<td>90.4</td>
<td>6.2</td>
<td>0.069</td>
</tr>
</tbody>
</table>

Define checks on data (later: using checks from Blaise)

- \( \frac{\text{abs(profit97 - profit98)}*2}{(\text{profit97} + \text{profit98})} < 0.1 \) --> percentage
- \( \text{Filling(NoPersons)} < 0.3 \) (non-response) --> signalling flag

<table>
<thead>
<tr>
<th>Mean</th>
<th>profit97</th>
<th>profit98</th>
<th>Δ profit</th>
<th>Filling &lt; 0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>234</td>
<td>250</td>
<td>2.3%</td>
<td></td>
</tr>
<tr>
<td>Group 2</td>
<td>456</td>
<td>503</td>
<td>14.7%</td>
<td></td>
</tr>
</tbody>
</table>
Scatter plot with condition

(Points not complying to the conditions are green)
Micro level: record editing

- Identify suspicious records either automatically or manually
- Sort suspicious records according to some outlier index, e.g. mahalanobis distance
- Edit records manually; worst cases first
Micro editing

- Directly: click on suspicious point in xy-plot and edit it
  - now:
    - copy OLEDB --> Blaise
    - call RunDEP(...) in BlTools.dll
    - copy Blaise --> OLEDB
  - later:
    - use DEC with existing IBlaiseDatabase reference
MacroView and Slice:
The building blocks of macroview

- Under the hood the macroview application consists of several modules; these modules can and will be used as building blocks in SLICE.
Slice: a decomposed macroview as an example
Slice

- Overall framework for editing/imputation and display modules, based on COM technology

- *Data source independence*: slice record sets
- *Small building blocks* with just 1 function: (COM) idea, LEGO bricks: flexibility

- Data exchange between blocks based on SLICE records, data description uses Blaise meta
- Usable in many environments because of COM: VB, VC++, Delphi, Manipula,...

- Data from / to: Blaise, OLEDB databases, ...

- Currently available: CherryPi and Imputation
Future of macroview (1):

- new aggregate functions for demography changes:
  - import/export(data source1, data source 2)

<table>
<thead>
<tr>
<th></th>
<th>Year98</th>
<th>Year99</th>
<th>Import(..)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>100.2</td>
<td>140.2</td>
<td>4</td>
</tr>
<tr>
<td>Group 2</td>
<td>90.4</td>
<td>88.3</td>
<td>0</td>
</tr>
</tbody>
</table>
Future of macroview (2):

- one more intuitive dialog for the definition of integer/float (time/date) variable types
- Manipulation of defined ranges
Future of macroview (3):

- More data sources:
  - all data sources (Di) must contain the grouping fields (G)
  - other data sources with detail records: join fields (J) must be supplied
  - in case of t-1, t-2, ...: often just aggregates; table meta: Cristal

![Diagram showing data sources and fields]

- D1: 
  - G
  - J
  - D1

- D2: 
  - G
  - J
  - D2

<table>
<thead>
<tr>
<th>G1</th>
<th>V1</th>
<th>V2</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gn</th>
<th>...</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Future of macroview (4):

- plots of data at aggregated level, e.g. histogram of average profit for some or all groups

<table>
<thead>
<tr>
<th>mean</th>
<th>year98</th>
<th>year99</th>
<th>import(=)</th>
</tr>
</thead>
<tbody>
<tr>
<td>group 1</td>
<td>100.2</td>
<td>140.2</td>
<td>4</td>
</tr>
<tr>
<td>group n</td>
<td>90.4</td>
<td>88.3</td>
<td>0</td>
</tr>
</tbody>
</table>

- more than one aggregate function active in grid

<table>
<thead>
<tr>
<th>mean</th>
<th>std</th>
<th>year98</th>
<th>year98</th>
<th>year99</th>
</tr>
</thead>
<tbody>
<tr>
<td>group 1</td>
<td></td>
<td>10.2</td>
<td>100.2</td>
<td>140.2</td>
</tr>
<tr>
<td>group n</td>
<td></td>
<td>9.4</td>
<td>90.4</td>
<td>88.3</td>
</tr>
</tbody>
</table>
Future of macroview (5):

- allow classification variable as a grouping variable (SBI,..)
- box-whisker, …, plots
- Integration with Blaise:
  - Macroview as a whole part of the Blaise shell
  - Aggregation and robust outlier detection (Kosinsky) as slice modules; available via Manipula
Slice in more detail
MacroView and Slice:
The building blocks of macroview
Slice modules:

- Receives and sends so-called Slice records
- Has no knowledge of the origin or destination of the records
- Has (in principle) one function
- Most modules will be non-interactive
Define/select variables to observe

- Derived variables can also be conditions, e.g.
  - Filling(NoPersons) > 0.3 (non-response) --> signalling flag
  - abs(profit97 - profit98)*2/(profit97 + profit98) < 0.1 --> percentage
  - alternative aggregate function: std(a)/mean(a) or just mean(a)
  - .
Storage

- Storage is done using the IRegister component in BlRegA.dll
- Undo/Redo ??
Blaise OLEDB Information (.boi) file

- Reference to Blaise meta file (.bmi) file
- Reference to an OLEDB provider and table, e.g.
  - provider = C:\MyDatabase.mdb (Access) , table = Customer
- Mapping between OLEDB fields and Blaise fields:
  - Customer.Name (String) <-> Name (STRING[20])
  - Customer.Type (Integer) <-> Type (AgeClass : (Rich (1), Medium (2), Poor (3) ) )
SiAggregate

Datasource (.boi)

Income, Age,...

Grouping:
  - GetQuery
  -

Integer Grouping
Enum Grouping
Classification Grouping
SiAggregate

Data → Stratum → StratumDef

Stratum:
- Cells
  - Var
    - Aggregates
  - VarRanges
    - Range
    - Var
Slice records:

- Database-independent and simple recordset
- Records can come from/go to any OLEDB or Blaise source
- Will handle the distribution to multiple modules
- Will take care of buffering of records, if necessary

The internal structure of a slice record
Using the slice modules:

- Both the modules and the Slice recordset can be used from any programming language that supports the use of COM
- Modules under construction:
  - CherryPi: automatic error detection
  - SilImputeXXX: several imputation modules
Near future:

- Com object for the Slice records
- Modules: CherryPi and SiRegressionImputation

Future:

- More modules (Aggregation, more imputation methods, ...)
- Visual prototyping tool
Visual inspection methods:

- Show Mahalanobis distances for selected recordset
- Show outliers (based on Kosinski algorithm) in datagrid or in scatterplot
- Mark outliers by hand in a scatterplot
- Clicking outlier point(s) allows editing of the record (using the Blaise DEP)
- Visual presentation
  - Flagged aggregates
  - List of sorted records
  - Scatter plot
  - Histogram
An example: the CherriPi module:

- Input: Recordset and a set of Edit Rules
- Output: Copy of input recordset with some fields marked ‘erroneous’
- Based on edit rules, e.g.
  
  \[
  \text{profit + costs} = \text{turnover} \\
  \text{costs} - 0.5 \times \text{turnover} > 0
  \]
- Automatic localization of erroneous records
- Localize faulty item(s) in erroneous record: e.g. algorithm as implemented by Cherry Pi (based on the Fellegi-Holt paradigm)
## Multiple datasources (e.g.: t, t-1)

<table>
<thead>
<tr>
<th></th>
<th>Situation 1</th>
<th>Situation 2</th>
<th>Situation 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupled</td>
<td>fully</td>
<td>partially</td>
<td>not</td>
</tr>
<tr>
<td>Grouping</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Aggregated</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Remark</td>
<td></td>
<td></td>
<td>Cristal</td>
</tr>
</tbody>
</table>
Cristal:

- **Dimensions** = Variable
- **Observation Item** = Variable + aggregation method
- **Hierarchy** = Variable + range:
  - **Classification item** = range
- **Data point** = 1 cell in grid
  = classification item x observation item