Improvements in editing methods and processes for use of Value Added Tax data in UK National Accounts

Martina Portanti and Robert Breton
Office for National Statistics, UK
Overview

- ONS use of VAT turnover data
- Data challenges
- Current methods micro editing and processing
- Recommended methods improvements
- System
- Timeline
ONS use of VAT data

- Historically VAT used as a source for the Business register
- 2013 ESSNet on Administrative data provided opportunity to explore methods to use VAT turnover data for producing business statistics
- Ad-hoc system built to process data for further research/exploration
- Since then, attempt to re-use system for statistical production purposes
- Limitations in both methods and system
Data challenges

1. Data needs to be calendarised to a monthly series

2. Data needs to be linked and apportioned to Reporting Unit level to ensure correct industrial classification and identify business employment sizeband
Current process

Three editing stages
1. Take on
2. After apportionment
3. After calendarisation

Structural validation at stage 1

Content validation at stage 2 and 3
1. Thousands pounds rule
2. Quarterly pattern rule
3. Suspicious turnover rule
Issues

1. Position of editing modules
2. Validation methods
3. Cleaning methods
4. Macro-editing stage
Target process

Key changes:

1. Content validation module at VAT unit level
2. Selective editing stage
3. No editing after calendarisation
1. Content validation module at VAT unit level

- “The early, the better”
- £000 rule and quarterly pattern rule
- Error detection
- Error correction may still be required at RU level
2. Selective editing stage

- Clerical investigation of influential cases earlier in the process
  - Accept
  - Impute
  - Manual construction
- Decrease macro-editing effort
- Identify genuine business changes
- Improve knowledge of new data source
- Improve micro-editing methods
3. No editing after calendarisation

- Ensure impact of editing vs calendarisation can be easily separated
- Allow more sophisticated calendarisation methods to be applied
- Streamline data processing system
Current system

- SAS-macro based system utilising 20 main macros
- Around 200 files created to process each data period
- 1 day processing
- Lack of interactive features to query and clean data
- Lack of flexibility to test different methods/sequencing
Ongoing system development

- Distributing computing technology
  - SPARK
  - Jupyter notebook
  - Oozie
- Processing time down to 5 minutes to process 4 years of data
- Agile approach with collaborative work across methodologists, data scientists and IT developers
This notebook will aim to understand whether all the reporting units in the matched and linked data fit into a cell in one of the three selection parameters files for inudexes '009', '033' and '529'.

```python
from pyspark.sql import HiveContext
from pyspark.sql.functions import udf, col
import pandas as pd
import jupyterutils.common as common
import pyspark.sql.functions as pyspark_functions
HIVE_CONTEXT = HiveContext(sc)
# required to read from partitioned tables
from collections import OrderedDict
from jupyterutils.graphs import build_bar_graph
from bokeh.plotting import figure, show, output_notebook, vplot, ColumnDataSource
from bokeh.models import HoverTool, BoxSelectTool, WheelZoomTool, FixedTicker, LinearColorMapper, ResetTool, SaveTool, B
from bokeh.palettes import Spectral11
import sys
from bokeh.charts import Bar, Histogram, HeatMap
from bokeh.layouts import row
output_notebook()
```

Create OrderedDict to store all counts during this notebook.

```python
In [2]: count_dict = OrderedDict()
```

Load in the reporting unit table we will use for this analysis, which is the rep._unit_2016_03_synth. It is worth noting that during this run (15/03/2017) there is a noticeable error in the frost2007 column for this reporting unit table as around 37000 only have 4 digits. This should not be the case and should be improved for a future iteration.

```python
In [3]: rep_units_synth = HIVE_CONTEXT.sql('''SELECT rusef, frost2007, region, rusefempent FROM matching.repunits_201603_synth_correct''')
```

We store a count of the reporting units in the table which we start with as a reference later in the notebook.

```python
In [4]: count_dict['total_reporting_units'] = rep_units_synth.count()
```

Next we calculate the number of alive reporting units. We know that in this project we are not interested in dead reporting units.

```python
In [5]: rep_unit_alive = HIVE_CONTEXT.sql('''SELECT rusef, frost2007, region, rusefempent FROM matching.repunits_201603_synth_correct where trim(Deathdate) = ""''
count_dict['total_alive_units'] = rep_unit_alive.count()
```

```python
In [6]: common.show_pretty(rep_unit_synth,5)
```
We carry out analysis of the three columns for each rur. We understand that any reporting units with region "YY" or region "ZZ" may not find a cell in the selection parameters. We will not exclude these reporting units at this stage.

```
In [7]: common.show_pretty(rep_unit_alive.groupby('region').count())
```

```
Out [7]

<table>
<thead>
<tr>
<th>region</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>GF</td>
<td>74216</td>
</tr>
<tr>
<td>GG</td>
<td>12308</td>
</tr>
<tr>
<td>AA</td>
<td>49853</td>
</tr>
<tr>
<td>ZZ</td>
<td>2662</td>
</tr>
<tr>
<td>HH</td>
<td>361932</td>
</tr>
<tr>
<td>BA</td>
<td>18143</td>
</tr>
<tr>
<td>BB</td>
<td>165284</td>
</tr>
<tr>
<td>JG</td>
<td>308195</td>
</tr>
<tr>
<td>DC</td>
<td>132617</td>
</tr>
<tr>
<td>WW</td>
<td>76725</td>
</tr>
<tr>
<td>KI</td>
<td>175215</td>
</tr>
<tr>
<td>ED</td>
<td>133146</td>
</tr>
<tr>
<td>XX</td>
<td>144903</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>PE</td>
<td>135308</td>
</tr>
<tr>
<td>YY</td>
<td>59314</td>
</tr>
</tbody>
</table>
```

By describing the 'fromempmen' column, which tells us the frozen employment values, we see that the minimum value is -1.

```
In [8]: common.show_pretty(rep_unit_alive.describe('fromempmen'))
```

```
Out [8]

<table>
<thead>
<tr>
<th>summary</th>
<th>fromempmen</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>1979207</td>
</tr>
<tr>
<td>mean</td>
<td>20.33005689465449</td>
</tr>
<tr>
<td>std</td>
<td>262.301593873096</td>
</tr>
<tr>
<td>min</td>
<td>-1</td>
</tr>
<tr>
<td>max</td>
<td>999</td>
</tr>
</tbody>
</table>
```

This is an unexpected value for fromemp men therefore we should investigate this further.

```
In [9]: common.show_pretty(rep_unit_alive.where(rep_unit_alive['fromempmen'] == -1))
```

```
Out [9]

<table>
<thead>
<tr>
<th>ruref</th>
<th>from2007</th>
<th>region</th>
<th>fromempmen</th>
</tr>
</thead>
</table>
```
It has been noted that the x-axis is not clear and readable but it is understood that the graphs may have some information needed.

```python
In [21]: gb_six = left_ever_rurefs.groupby('frosic2007').count().orderBy('frosic2007')

In [22]: graph = build_bar_graph(gb_six, xaxis='frosic2007', yaxis = 'count', title = 'Count of rurefs not in a cell by SIC Code', xaxislabel = 'frosic2007', yaxislabel = 'count', data_labels = ['frosic2007', 'count'])
```

Cell Mapping 023.

By joining the reporting unit dataframe to the selection parameters file (with Composite SICs included) for Impode '023', we can see which of the reporting units find a cell.

```sql
In [23]: rep_cells_023 = HIVE_CONTEXT.sql('SELECT s.cell_no_def, rep.ruref as ruref, rep.frosic2007, rep.frcomponent, rep.region, s.lower_class, s.upper_class, s.upper_region, s.lower_region, s.lower_site, s.upper_area')
```
Oozie workflow
Timeline

- July 2017 – delivery of a VAT turnover processing system
- July 2017-December 2017 – National Accounts testing within NA systems
- End of 2017 – publication of Q3 quarterly national accounts estimates including VAT data for the first time
Thank you

Contact details

• Martina Portanti
  martina.portanti@ons.gov.uk
  Head of Processing, Editing and Imputation (Business surveys)
  Survey Methodology and Statistical Computing division

• Robert Breton
  robert.breton@ons.gov.uk
  Head of Data Sources Data Science
  Economic Statistics Data Sources