Office for National Statistics

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

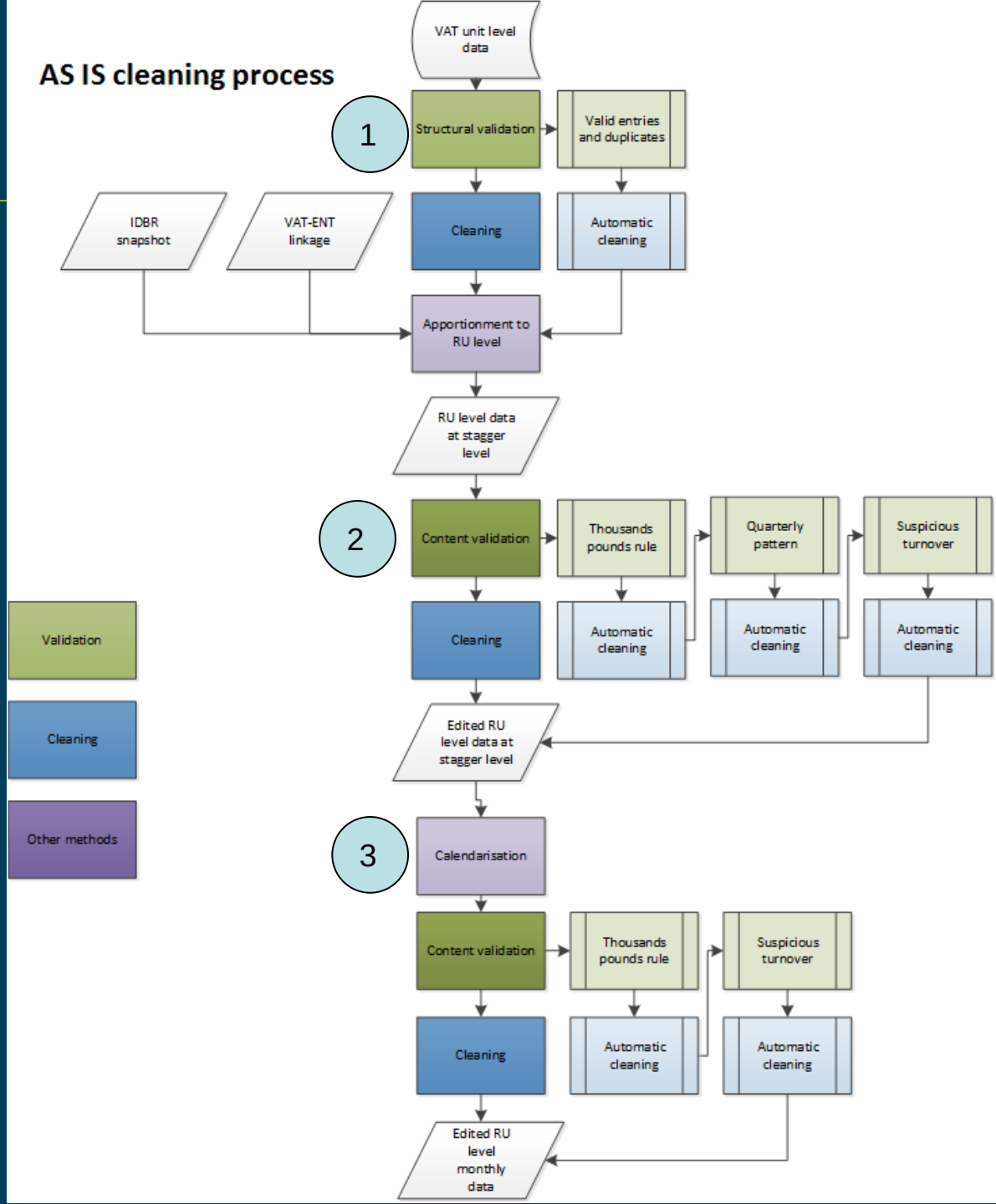
Current process

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



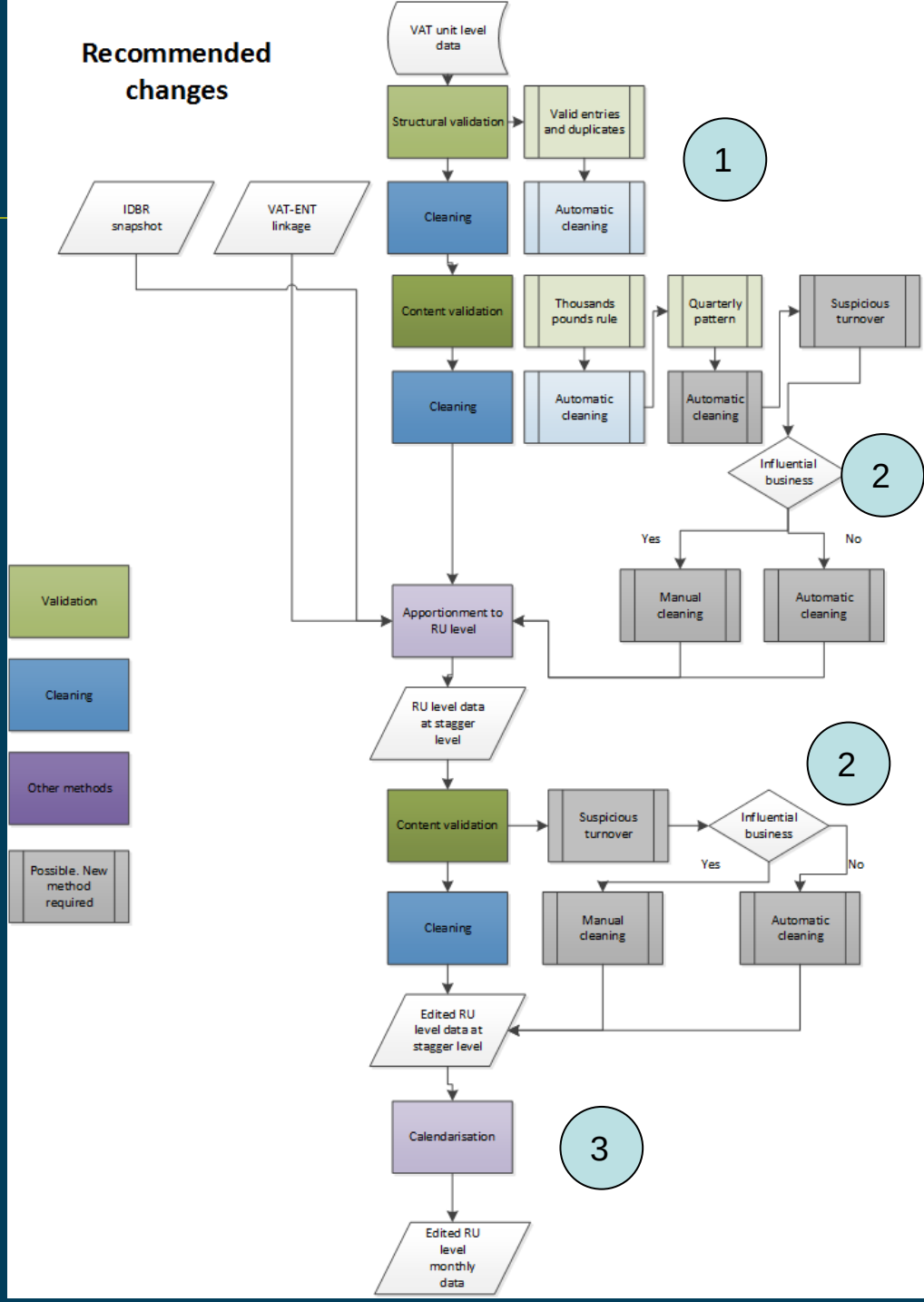
Issues

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



Recommended improvements/1

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

Recommended improvements/2

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

Recommended improvements/3

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

Jupyter notebook/1

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 inqcodes '009', '023' and '228'.

In [1]:
```
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
HIVE_CONTEXT.setConf("spark.sql.hive.convertMetastoreParquet","false")
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
oxZoomTool, PanTool, Span, Range1d, DatetimeTickFormatter, SingleIntervalTicker
from bokeh.palettes import Spectral11, d3
import sys
from bokeh.charts import Bar, Histogram, HeatMap
from bokeh.layouts import row
output_notebook()
```

BokehJS successfully loaded.

Create OrderedDict to store all counts during this notebook.

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 noticable error in the frosic2007 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.

In [3]:
```
rep_unit_synth = HIVE_CONTEXT.sql('''SELECT ruref, frosic2007, region, froempment FROM vatmatching.repunit_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.

In [4]:
```
count_dict['total_reporting_units'] = rep_unit_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.

In [5]:
```
rep_unit_alive = HIVE_CONTEXT.sql('''SELECT ruref, frosic2007, region, froempment FROM vatmatching.repunit_201603_synth
_correct where trim(deathdate) = "" ''')
count_dict['total_alive_units'] = rep_unit_alive.count()
```

In [6]:
```
common.show_pretty(rep_unit_synth,5)
```

Jupyter notebook/2

We carry out analysis of the three columns for each ruref. 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]:

	region	count
0	GF	74215
1	GG	123058
2	AA	49353
3	ZZ	2662
4	HH	361932
5	BA	18143
6	BB	165284
7	JG	308195
8	DC	133617
9	WW	76725
10	KJ	175219
11	ED	133148
12	XX	144903
13		11
14	FE	153508
15	YY	59314

By describing the 'froempment' 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('froempment'))
```

Out[8]:

	summary	froempment
0	count	1979287
1	mean	20.33050689465449
2	stddev	602.3015983870386
3	min	-1
4	max	999

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

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

Out[9]:

	ruref	frosic2007	region	froempment

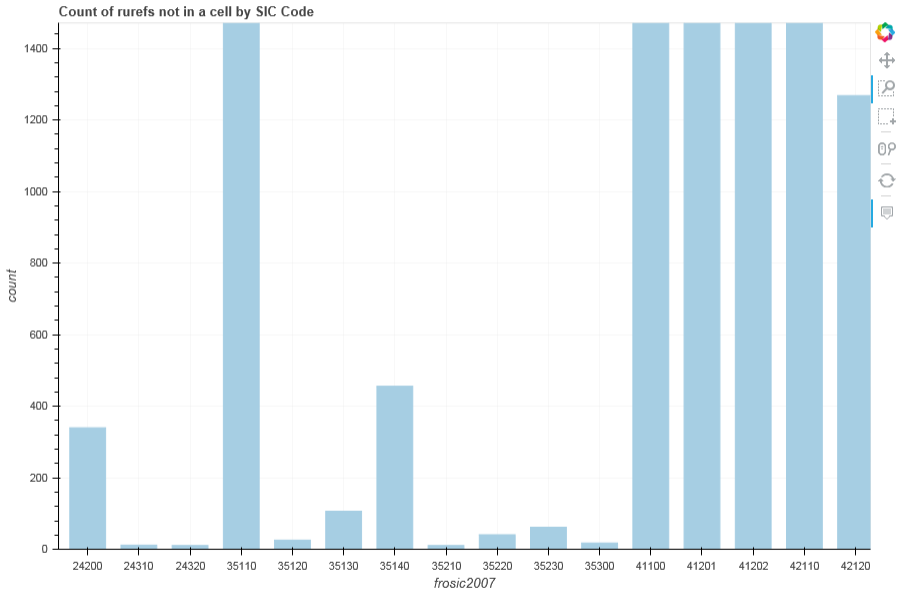
Jupyter notebook/3

C:\Users\portam\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content | Reggie | Files · master · Enda Farre... | CIRCABC - Task force - V... | circabc.europa.eu | CIRCABC - Login | onsvmp_59 | ONSVMP_90_do_repo...

> 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.

In [21]:
```
gb_sic = left_over_rurefs.groupBy('frosic2007').count().orderBy('frosic2007')
```

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



Cell Mapping 023.

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

In [25]:
```
rep_cells_023 = HIVE_CONTEXT.sql('''SELECT s.cell_no_def, rep.ruref as ruref, rep.frosic2007, rep.froempment, rep.regio
n, s.lower_class, s.upper_class, s.upper_region, s.lower_region, s.lower_size, s.upper_size
```

11:33
20/04/2017

Oozie workflow

http://cdhmn-d03-03.ons.statistics.gov.uk:8888/oozie/editor/workflow/edit/?workflow:

Accessing the 'Non Live Busin... | Hue - Workflow Editor

Query Editors | Notebooks | Data Browsers | Workflows | Search | File Browser | Job Browser | brownb

Oozie Editor | Workflows | Coordinators | Bundles

Sprint 5 PySpark - 4

Runs 4 python spark jobs.

Execute Spark2 - half

Execute-spark2.py yarn-cluster cluster
hdfs://dev3/user/hue/oozie/workspaces/hue-oozie-1481636788.46/lib/spark2_edited.py

Execute Spark3 - add 10

spark3 yarn-cluster cluster
hdfs://dev3/user/hue/oozie/workspaces/hue-oozie-1481636788.46/lib/spark3.py main class

Execute Spark4a counts

spark4a yarn-cluster cluster
hdfs://dev3/user/hue/oozie/workspaces/hue-oozie-1481636788.46/lib/spark4a.py main class

Execute Spark4b totals

spark4b yarn-cluster cluster
hdfs://dev3/user/hue/oozie/workspaces/hue-oozie-1481636788.46/lib/spark4b.py main class

HDFS Fs

Move
VATOut3 to VATOut3_${wf:id()}_${replaceAll(timestamp(),':','')}
VATOut2 to VATOut2_${wf:id()}_${replaceAll(timestamp(),':','')}
VATOut4a to VATOut4a_${wf:id()}_${replaceAll(timestamp(),':','')}
VATOut4b to VATOut4b_${wf:id()}_${replaceAll(timestamp(),':','')}

14:20
20/02/2017

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