Consistently perturbing tables

Introducing R package cellKey
What (in a nutshell)

- Presenting R package cellKey (github.com/sdcTools/cellKey)
- Developed within project “Open Source tools for perturbative confidentiality methods”
  - Thx to all colleagues from Germany, Netherlands, France, Finland, Slovenia, Finland and Hungary
- The tool implements the (enhanced) cell-key method
  - originally proposed by ABS
  - post-tabular method to protect statistical tables
  - enhanced/modified within this project
Features:

- **Starting point:** microdata
  - important: assignment of record keys (rkeys)
  - allows to easily define (complex) tables
  - allows to perturb count- and magnitude tables using cell keys (ckeys) derived from rkeys

- (Sampling)weights are supported
- Allows for different perturbation parameters by variable
- many auxiliary functions (e.g key generation) available
The method: Starting from micro data

- **Idea**: describing the general idea of the method (easy)
- Post-tabular method when other methods (e.g. suppression) are not viable
- starting from a micro data set `md` (nrows: 4580)

```r
print(head(md))
```

```
## sex age high_inc w savings rkey
## 1: male ag3 0 70 12 0.8027299
## 2: female ag3 0 99 28 0.6282568
## 3: male ag1 0 58 550 0.3713780
## 4: male ag1 0 23 870 0.2507976
## 5: male ag4 1 38 20 0.9165093
## 6: female ag3 0 93 102 0.8307836
```
The method: Deriving a cell key

- Identify contributors for a specific cell:

<table>
<thead>
<tr>
<th></th>
<th>sex</th>
<th>age</th>
<th>high_inc</th>
<th>w</th>
<th>savings</th>
<th>rkey</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>male</td>
<td>ag6</td>
<td>0</td>
<td>29</td>
<td>771</td>
<td>0.11134575</td>
</tr>
<tr>
<td>2</td>
<td>male</td>
<td>ag6</td>
<td>0</td>
<td>90</td>
<td>767</td>
<td>0.75515496</td>
</tr>
<tr>
<td>3</td>
<td>male</td>
<td>ag6</td>
<td>0</td>
<td>38</td>
<td>821</td>
<td>0.56953559</td>
</tr>
<tr>
<td>4</td>
<td>male</td>
<td>ag6</td>
<td>0</td>
<td>26</td>
<td>41</td>
<td>0.45790072</td>
</tr>
<tr>
<td>5</td>
<td>male</td>
<td>ag6</td>
<td>0</td>
<td>59</td>
<td>281</td>
<td>0.16707917</td>
</tr>
<tr>
<td>6</td>
<td>male</td>
<td>ag6</td>
<td>0</td>
<td>39</td>
<td>558</td>
<td>0.05877474</td>
</tr>
<tr>
<td>7</td>
<td>male</td>
<td>ag6</td>
<td>0</td>
<td>81</td>
<td>371</td>
<td>0.06648822</td>
</tr>
</tbody>
</table>

- **ckey**: Sum of all contributing record keys and taking the fractal part

\[ \rightarrow \text{the cell keys are } \sim \in [0, 1) \]

| [1] | 0.1862791 |

- **Consistency**: the same set of units returns the same ckey
- **Perturbation**: find a perturbation value based on the ckey
**The method: Perturbing the cell**

- **Trivial**: using the `ckey` as seed when sampling a perturbation value

```r
pval <- function(ckey) { set.seed(ckey); sample(-10:10, 1) }
```

- **Perturbation**: we use the noise value to perturb the cell count

```r
result <- sum(contributors$w) + pval(ckey); result
```

```r
## [1] 365
```

- **Task**: Finding perturbation values → make use of a perturbation table

www.statistik.at
The ptable package (1)

- The CKM (and thus also cellKey) depends on perturbation tables
- These so-called ptables encompass noise values and the corresponding probabilities
- There are ptables for both: frequency and magnitude tables
- The ptables need to be designed such that
  - no bias is introduced
  - the variance of the noise is fixed
  - in case of frequency tables:
    - no negative counts will appear
    - zero counts remain zeroes
ptables can be computed with R package ptable (github.com/sdcTools/ptable)

The ptable results from the solution of non-linear equation systems that are defined by contraints and boundary conditions relying on a maximum entropy optimization

Installation and Loading as easy as:

```r
remotes::install_github(
  repo = "sdcTools/ptable",
  dependencies = TRUE)

library(ptable)
```
The package allows for a couple of arguments - used for the constraints of the optimization instance - when designing the ptable:

- **D** maximum noise
- **V** variance
- **pstay** probability that a frequency count remains unchanged/unperturbed
- **js** small counts can be blocked (i.e. they will not appear in the protected table)
- ... and many more ...

**Vignette** gives an introduction with examples: call `pt_vignette()`
Example: Design the ptable for a frequency table with maximum noise $D=2$, a fixed variance $V=1.05$, counts of 1 shall not appear in the protected tables and the probability that a frequency count is set to 50%.

```r
ptab <- ptable::create_cnt_ptable(D = 2, V = 1.05, js = 1, pstay = 0.5, mono = c(T, T, F, T))
```
Example (continued)

```r
# Evaluate the result
ptab@empResults
```

<table>
<thead>
<tr>
<th></th>
<th>i</th>
<th>p_mean</th>
<th>p_var</th>
<th>p_sum</th>
<th>p_stay</th>
<th>iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
<td>1</td>
<td>1.0000</td>
<td>0</td>
</tr>
<tr>
<td>2:</td>
<td>1</td>
<td>0</td>
<td>1.05</td>
<td>1</td>
<td>0.0000</td>
<td>1</td>
</tr>
<tr>
<td>3:</td>
<td>2</td>
<td>0</td>
<td>1.05</td>
<td>1</td>
<td>0.5557</td>
<td>1</td>
</tr>
<tr>
<td>4:</td>
<td>3</td>
<td>0</td>
<td>1.05</td>
<td>1</td>
<td>0.2881</td>
<td>6</td>
</tr>
<tr>
<td>5:</td>
<td>4</td>
<td>0</td>
<td>1.05</td>
<td>1</td>
<td>0.5000</td>
<td>1</td>
</tr>
</tbody>
</table>
Example (continued)

```r
# Look at the ptable
head(ptab@pTable)
```

```r
## i j p v p_int_lb p_int_ub type
## 1: 0 0 1.00000000 0 0.0000000 1.0000000 all
## 2: 1 0 0.50833333 -1 0.0000000 0.5083333 all
## 3: 1 2 0.47500000 1 0.5083333 0.9833333 all
## 4: 1 3 0.01666667 2 0.9833333 1.0000000 all
## 5: 2 0 0.16155827 -2 0.0000000 0.1615583 all
## 6: 2 2 0.55565037 0 0.1615583 0.7172086 all
```
Example (continued)

```r
plot(ptab, type = "p") # Plot the ptable (perturbation panel)
```

## Perturbation Panel

Obtaining perturbation tables

slide 13  |  October 2019
Getting started with the cellKey package

Installation
- directly from the [github.com/sdcTools/cellKey](https://github.com/sdcTools/cellKey) (not yet on CRAN)

(online) Dokumentation: [sdctools.github.io/cellKey](https://sdctools.github.io/cellKey)
- includes a package vignette

defining hierarchies
- cellKey uses functionality from [sdcHierarchies](https://sdctools.github.io/sdcHierarchies)
- hierarchies can be created
  - indirectly (from a character vector)
  - directly (by adding nodes to a tree)
Prerequisites: micro data, hierarchies, optionally count and continuous variables, weights and record keys

Dimensions

dims <- list(
  sex = hier_create("total", nodes = c("male", "female")),
  age = hier_create("total", nodes = paste0("ag", 1:6)))

Table instance

tab <- ck_setup(x = md, rkey = "rkey", dims = dims,
                w = "w", countvars = c("high_inc"), numvars = "savings")
objects created with `ck_setup()` have their data and methods bundled together

methods can be called with the following syntax

```
obj$method(...)  
```

for example (no perturbed results yet):

```
head(tab$freqtab(v = "total"))  
```

<table>
<thead>
<tr>
<th></th>
<th>sex</th>
<th>age</th>
<th>vname</th>
<th>uwc</th>
<th>wc</th>
<th>puwc</th>
<th>pwc</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>total</td>
<td>total</td>
<td>total</td>
<td>4580</td>
<td>274917</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>2</td>
<td>total</td>
<td>ag1</td>
<td>total</td>
<td>1969</td>
<td>119138</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>3</td>
<td>total</td>
<td>ag2</td>
<td>total</td>
<td>1143</td>
<td>68337</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>4</td>
<td>total</td>
<td>ag3</td>
<td>total</td>
<td>864</td>
<td>51841</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>5</td>
<td>total</td>
<td>ag4</td>
<td>total</td>
<td>423</td>
<td>24759</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>6</td>
<td>total</td>
<td>ag5</td>
<td>total</td>
<td>168</td>
<td>10105</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>
Attaching parameters to variables:

- for count variables: `ck_params_cnts()`
- for continuous variables: `ck_params_nums()`

Example: Parameters for a count variable

```r
library(ptable)
cnt_params1 <- ck_params_cnts(ptab = create_cnt_ptable(D = 8, V = 3, js = 2))
cnt_params2 <- ck_params_cnts(ptab = create_cnt_ptable(D = 10, V = 10, js = 5))
```
Attaching parameters to variable(s)

- `params_cnts_set()` / `params_nums_set()`

```
`tab$`params_cnts_set(cnt_params1) # default: all vars
```

## --> setting perturbation parameters for variable 'total'

## --> setting perturbation parameters for variable 'high_inc'

- for specific variables (resetting previous parameters)

```
`tab$`params_cnts_set(v = "high_inc", val = cnt_params2)
```

## --> replacing perturbation parameters for variable 'high_inc'
Perturbation: use the `perturb` method (for both count- and continuous variables)

```r
tab$perturb(c("total", "high_inc"))
```

```r
## Count variable 'total' was perturbed.
## Count variable 'high_inc' was perturbed.
```

Results: use the `freqtab()` and `numtab()` methods

```r
## sex age vname uwc wc puwc pwc
## 1: total total high_inc 445 26535 444 26475.3708
## 2: total ag1 high_inc 192 11881 190 11757.2396
## 3: total ag2 high_inc 123 6833 122 6777.4472
## 4: total ag3 high_inc 82 4980 78 4737.0732
## 5: total ag4 high_inc 34 1967 33 1909.1471
## 6: total ag5 high_inc 14 874 12 749.1429
```
CKM was extended quite a lot for magnitude tables

- different "starting values" for perturbation possible
- apply different pttables for cells with
  - an even/odd number of contributors
  - very small values
- allow for extra perturbation for sensitive cells
  - supp_{method}() namespace for identification

- distance/utility measures for count-variables (measures_cnts())
- perturbation parameters can be saved for reproducability (ck_write_yaml() and ck_read_yaml())
Future development:
- incorporating **feedback** from users
- add utility statistics and measures for continuous variables
- achieve full consistency with the implementation from τ-args

Summary
- **cellKey** allows to consistently perturb statistical tables
- it makes use of ptables created from the **ptable** package

Important Links
- **Documentation**: sdctools.github.io/cellKey/
- **Questions**: github.com/sdcTools/userSupport
- **Bug-Reports**: github.com/sdcTools/userSupport/issues
Consistently perturbing tables

Introducing R package cellKey