Validating and Extracting Information from Finnish and Swedish National Identification Numbers with the hetu and sweidnumbr Packages

by Pyry Kantanen, Jussi Paananen, Leo Lahti

Abstract National identification numbers (NIN) are a widely used method for uniquely identifying individuals and companies in Finland, Sweden, and many other countries. Their widespread use in both public sector registers and private sector customer data systems prompts the need for openly available methods and tools for NIN analysis and validation. The *hetu* and *sweidnumbr* R packages provide functions for extracting information from Finnish and Swedish NINs. The packages can also generate temporary but structurally valid random numbers for testing purposes. Finnish and Swedish NINs contain information about person's birth date and sex, an individual number and a control character that helps detect most common input errors and structural irregularities. Our packages contains comprehensive documentation and examples to make usage of the package easy for domain experts with non-technical backgrounds. While the new tools are relevant mainly in the Finnish and Swedish context, this package contributes to developing similar tools for other countries. From a wider persepective, both *hetu* and *sweidnumbr* R packages contribute to the growing toolkit of standardized methods for computational social science and register-based analytics.

Introduction

Reliably keeping track of individuals in a given territory has long been seen as an important feature of modern governance. The scope of this tracking has spread from disciplinary institutions, such as prisons, military units and schools to society at large (Foucault, 2009). We can see the results of this development today in the form of different *national identification number* (NIN) systems implemented around the world.

An ideal personal identification system should strive to be both unique and "self-same" over time. Self-sameness refers to a degree of immutability that allows organizations to identify and reidentify a person over time. A combination of attributes[1] such as name, occupation and address would probably form a unique identifier even in relatively large crowds, but such attributes might not stay the same over time. (Brensinger and Eyal, 2021). In everyday life settings, a combination of personal attributes, such as name, occupation and address may be sufficiently *unique* to identify individuals even in large groups of people. Such an approach has its limitations. The problem with such composite keys is, however, that parts that make the composite key might not be permanent, making it impossible to compare different data points in the same dataset or combining data from different datasets.

According to Alterman (as cited in Ajana, 2013, 8) a distinction can be made between *biocentric data* and *indexical data*. The former is biometric data connected to the *body* of the individual whereas the latter has no distinguishable relation to the individual, physiologically, psychologically or otherwise. An example of biocentric data could be a fingerprint or an iris scan and an example of indexical data could be a randomly assigned number from which nothing could be deduced.[2]

In casual settings the combination of person's name and physiological characteristics (or *biometric identifiers*) are enough to identify friends, acquaintances and strangers from one another. Names can change over time and even fingerprints and iris pattern samples need to be refreshed from time to time, making them poor from self-sameness perspective. Interestingly, Lauer (2017) and Rule (1973) (as cited in Brensinger and Eyal, 2021, 31–32) noted that in the 1960's American credit bureaus and other organizations struggled with this same problem having used a combination of attributes to identify people in their registers and needed to find a better way to identify individuals. In a moment

[1] Think of SQL's *composite key*

[2] Brensinger and Eyal (2021) discuss the concept of *dividuals*, manufactured objects that represent the living individual: address, fingerprints, name and so on. These dividuals need to go through the process of disembedding, standardization and reembedding to be useful. Disembedding means data gathering (taking a fingerprint sample), standardization means making the disembedded transcription into a standardized digital sample that can be easily compared with other similar samples and reembedding means linking these standardized records back to their actual flesh-and-blood counterparts. Without a way to reembed a huge and well standardized archive of fingerprints back to the population it is essentially useless. This is also a reason why biometric samples such as iris scans or fingerprints can never replace a primary keys in databases.

that may be described as *function creep* they settled for using the *social security number* (SSN) as the single identifying number[3], horizontal and vertical information sharing possible.

For a number of reasons, many identification numbers are not just random strings. The American SSN originally contained information about the person's birth year and where the number was first registered (Brensinger and Eyal, 2021, 32) whereas Nordic countries' national identification numbers often contain (or used to contain) information about the individual's birth date and sex (Watson, 2010; Salste, 2021). Therefore they may be seen as biocentric data rather than pure indexical data, making them more sensitive to handle.

In the Nordic countries comprehensive national identification number systems were developed and implemented from the 1940's to 1960's (Watson, 2010). In Sweden, the personal identity number was introduced in 1947 as a way to identify individuals for tax purposes. The personal identity number consisted both of the date of birth and an additional three-digit birth number that together uniquely identified every individual in Sweden. In 1967 a control digit to easier identify incorrect entries was added to finalize the full Swedish personal identity number (or *personnummer*) (Åke Johansson, 2003; Statistics Sweden, 2016).

The Finnish personal identity code has its roots in specialized employment pension number from 1962, which was gradually expanded to cover the whole population in the form of a social security number in 1964-1968. The structure of pension number was designed by mathematician Erkki Pale with punch card machines in mind (probably influenced by the Swedish födelsenummer or birth number). This design has proved to be resilient and with some tweaks it continues to be used, with the modern iteration being called a *personal identity code* (In Finnish: henkilötunnus, or *hetu* for short, hence the name of the package) (Salste, 2021). Because of this, we will be using the term personal identity code (PIC) when referring to the Finnish national identification number.

The use of national identification numbers as widespread both in Finland and in other countries. In Finland the adoption of personal identity codes in 1969 has allowed the widespread use of secondary data sources[4]. One example of this is the use of Finnish Hospital Discharge Register (FHDR), which contains data on all inpatient hospital discharges that can be linked to other sources with personal identity code. According to Sund (2012) the proportion of erroneously inputed PICs or incomplete pseudo-PICs was initially high but fell rapidly as hospitals were specifically instructed to include it in discharge records starting from 1972. When FHDR diagnoses were compared to other sources, their quality was deemed to be at least satisfactory to good when limitations are taken into account.

In Sweden the PIC is currently used extensively in all parts of society, not only for taxation. It is used in education, for military service, in health care and by financial institutions, and insurance companies. The role of the PIC has also made it central to register-based research (Statistics Sweden, 2016).

Similar R packages **numbersBR** for Brazilian identity numbers for individuals, vehicles and organisations (Freitas, 2018), and **generator** for generating various types of Personally Identifiable Information (PII), such as fake e-mail addresses, names and United States Social Security Numbers (Hendricks, 2015).

Personal identity code generators and validators are nothing new. In fact, handling hetu codes seems to be a common entry-level task to familiarize new computer science students with regular expressions, dates, string subsetting and similar concepts. The reasoning behind packaging these functions as an R package is the same as stated in Wickham and Bryan's R Packages Introduction: code sharing, transparency, user manuals, semantic versioning and documenting changes between versions (Wickham and Bryan, 2022). In addition, there is of course intrinsic value in providing a solution to a common problem in the R language.

Working with personal identity codes

The method of validating and extracting information from identification numbers is manually doable and simple in principle but in practice becomes unfeasible with datasets larger than a few dozen observations. The hetu and sweidnumbr packages provides easy-to-use tools for programmatic handling of Finnish and Swedish personal identity codes and Business ID codes.[5] The packages utilizes R's efficient vectorized operations and is able to generate and validate over 5 million personal identity codes or Business Identity Codes in less than 10 minutes on a regular laptop. This covers the practical upper limit set by the current population of Finland (5.5 million people) and Sweden (10.35 million people), providing adequate headroom for handling of relatively large registry datasets.

[3] Think of SQL's *primary key*

[4] Secondary data: Data that has not been collected primarily for the purpose of a specific research question

[5] In Finnish: Yritys- ja Yhteisötunnus, or Y-tunnus for short, In Swedish: Organisationsnummer

The hetu package

Printing a table containing extracted information in a structured form can be done as follows:

```
x <- c("010101A0101", "111111-111C", "290201A010M")
hetu(x)
```

The results can be seen in Table 1. We can already see that there is a problem with the last personal identity code; 29th of February 2001 is not a valid date as 2001 is not a leap year.

	hetu	sex	p.num	ctrl.char	date	day	month	year	century	valid.pin
1	010101A0101	Female	010	1	2001-01-01	1	1	2001	A	TRUE
2	111111-111C	Male	111	C	1911-11-11	11	11	1911	-	TRUE
3	290201A010W	Female	010	M	<NA>	29	02	2001	A	FALSE

Table 1: Table printed with `hetu(pin = c("010101A0101", "111111-111C", "290201A010W"))`.

The generic way of outputing information found on individual columns is to use the standard hetu()-function with extract-parameter. For example:

```
hetu("010101A0101", extract = "sex")
[1] "Female"
hetu("010101A0101", extract = "date")
[1] "2001-01-01"
```

A valid extract parameter needs to be a column name in the table printed out by `hetu`-function. Most commonly used columns have their own function wrappers that are identical in output:

```
hetu_sex("010101A0101")
[1] "Female"
hetu_date("010101A0101")
[1] "2001-01-01"
```

By relying on **lubridate** we have also implemented a special function to calculate the age of individuals in years (default), months, weeks or days. The default option is to calculate age at the current moment but it is also possible to set a specific date at which the age is calculated.

```
hetu_age("010101A0101", date = "2004-02-01", timespan = "months")
The age in months has been calculated at 2004-02-01.
[1] 37
```

With the `hetu_diagnostic()` function we can take a closer look at diagnostics results of each personal identity code. The function prints information about 10 different checks done on the code. In this case we only want to check results related to validity of the personal number component, whether the control character is valid, whether the control character is correct and whether the date is valid.

```
diagnostics <- hetu_diagnostic("290201A010M")
diagnostics[,c("valid.p.num", "valid.checksum", "correct.checksum", "valid.date")]
```

The results of `hetu_diagnostic` can be seen in Table 2. We can see that the checksum character is valid, meaning that the character is either a number or an ASCII letter, excluding letters that could be mistaken for another character: G, I, O, Q or Z. Correctness of the checksum means whether the character is correct when date and personal number parts are concatenated together, divided by 31 and the remainder of this operator is used as a lookup value from a table containing valid control characters.

When data is inputed manually without validity checks, input errors can creep in. The control character in Finnish personal identity codes combined with validity checks in hetu package can help to catch the most obvious errors. In our example of we saw that the date is incorrect, but also the control character is incorrect, meaning that there has been an error in date or personal number input. Because the date is obviously wrong, we try three different dates and see if any of these are correct: 28th of February 2001, 29th of March 2001 and 29th of February 2000 (which was a leap year). Table 2 shows this comparison.

The `hetu` package can generate a large number of personal identity codes with the `rhetu` function. The date range of the generated identity codes can be changed with parameters, but it has a hard coded lower limit at the year 1860 and upper limit in the current date. It has been theorized that the

	hetu	valid.p.num	valid.checksum	correct.checksum	valid.date
1	290201A010M	TRUE	TRUE	FALSE	FALSE
2	280201A010M	TRUE	TRUE	FALSE	TRUE
3	290301A010M	TRUE	TRUE	FALSE	TRUE
4	290200A010M	TRUE	TRUE	TRUE	TRUE

Table 2: A subset of diagnostics of invalid personal identity code "290201A010M", its variations, and finally the correct personal identity code "290200A010M".

oldest individuals that received a personal identity code in 1960s were born in 1860s. Personal identity codes are never assigned beforehand and therefore it is impossible to have valid personal identity codes that have a future date.

The function can also be used to generate so called temporary identity codes. Temporary identity codes are never used as a persistent and unique identifier for a single individual but as a placeholder in institutions such as hospitals when a person does not have a Finnish personal identity code or it is not known. They can be identified by having a personal number in the range of 900-999.

Here is an example of generating 4 temporary PINs and checking their validity with `hetu_ctrl` function:

```
x <- rhetu(n = 4, p.male = 0.25, p.temp = 1.0)
x
[1] "160237-938R" "131166-950X" "151184-9241" "250104A954R"
hetu_ctrl(x, allow.temp = TRUE)
[1] TRUE TRUE TRUE TRUE
```

As mentioned earlier, our package also supports similarly generating and checking the validity of Finnish Business ID (BID) numbers. In addition we have added support for the less known Finnish Unique Identification (FINUID) numbers.[6] As the name implies, BIDs are used as unique identifiers for companies, organizations and other legal persons. Unlike personal identity codes, BIDs do not contain any information about the company. BIDs consist of a random string of 7 numbers followed by a dash and 1 control character, a number between 0 and 9. FINUID numbers are similar in the sense that they do not contain any biocentric data on the individual and are mainly used by government authorities in IT systems. FINUID numbers consist of 8 numbers and 1 control character calculated in the same way as in personal identity codes. Both of these ID numbers are an example of indexical data, as mentioned earlier.

```
bid_ctrl(c("0000000-0", "0000001-9"))
[1] TRUE TRUE

satu_ctrl("10000001N")
[1] TRUE
```

The `sweidnumbr` package

The `sweidnumbr` R package has similar functionality as the `hetu` package, but for Swedish PIC and with a slightly different syntax. As of now, the package has been downloaded roughly 30 000 times. The PIC below is an example PIC taken from The Swedish Tax Agency (2007).

Unlike the `hetu` package, the first step in `sweidnumbr` is to convert strings with different PIC formats or numeric variables into a `pin` vector using the `as.pin()` function

```
example_pin <- c("640823-3234", "6408233234", "19640823-3230")
example_pin <- as.pin(example_pin)
example_pin

[1] "196408233234" "196408233234" "196408233230"
Personal identity number(s)
```

The `pin` vector is a S3 object and can be checked is the `is.pin()` function.

```
is.pin(example_pin)

[1] TRUE
```

[6] in Finnish: Sähköinen asiointitunniste or SATU for short

This function only check that the vector is a `pin` object, but not if the actual PIC are valid. To check PIC using the control numbers we simply use the `pin_ctrl()` function.

```
pin_ctrl(example_pin)

[1]  TRUE  TRUE FALSE
```

Just as in the `hetu` package we can extract informations from the PIC with specialized functions. We can now use pin_birthplace(), pin_sex(), and pin_age() to extract information on birthplace, sex, and age.

```
pin_sex(example_pin)
[1] Male Male Male
Levels: Male

pin_birthplace(example_pin)
[1] Gotlands län Gotlands län Gotlands län
28 Levels: Stockholm stad Stockholms län Uppsala län ... Born after 31 december 1989

pin_age(example_pin)
[1] 55 55 55
pin_age(example_pin, date = "2000-01-01")
[1] 35 35 35
```

And, as with the `hetu` R package we can also generate, or simulate, PIC with the `rpin()` function. The same structure also holds for handling Swedish organizational number (SON), namely `as.oin()`, `is.oin()`, and `oin_ctrl()`. Unlike PIC, the `oin` contain information on the type of organization of a given SON.

```
oin_group(example\_oin)

[1] Aktiebolag
[2] Stat, landsting, kommuner, församlingar
[3] Ideella föreningar och stiftelser
3 Levels: Aktiebolag ... Stat, landsting, kommuner, församlingar
```

And just as in the `hetu` package and for PIC, we can simulate new SON using the `roin()` function.

```
roin(3)

[1] "964228-2835" "945463-1814" "736937-8059"
Organizational identity number(s)
```

Discussion

The hetu and sweidnumbr R packages provides a free and open source methods for validating and extracting data from a large number of Finnish and Swedish personal identity codes. While the package's target audience consists mostly of Finnish and Swedish users and people with particular interest in NIN systems around the world, the package makes a generic contribution in developing methodologies related to NIN handling in R. In the future, a more generic package or class structures could be useful to unify the handling of different NIN systems around the world, and might be worth exploring further.[7]

The origins of these packages can be traced to early 2010s when one curious individual wanted to analyze a large number of Finnish personal identity codes that were leaked to the internet by an anonymous hacker. The legality and morality of handling such dataset containing personal information was and is in a grey area at best. As developers of this package we cannot condone such activities, even if they are conducted out of curiosity and not ill intentions, but we acknowledge that we cannot prevent our users from doing that either.[8]

[7] Although in Unix philosophy one stated aim is "Write programs that do one thing and do it well" and "Write programs to work together" as opposed to writing one program that aims to do everything. The value in one generic program might mostly be related to sharing information and good coding practices among interested parties around the world.

[8] The use of "Good, not evil" license texts is thought to be problematic and against the ethos of open source software

We have acknowledged beforehand that random personal identity codes generated with the hetu package could theoretically be used for purposes such as synthetic identity fraud.[9] On the other hand it is important to note that such identity codes could also be created by hand as the same information that we have used in developing our algorithms is available at Finnish authorities' web page (Digital and Population Data Services Agency, 2022). Our package can be useful for many, and it does not make fraudulent activities significantly easier for malevolent individuals which is essential in judging the pros and cons of releasing this software to the public.

Similar data breaches have made people more wary of digital services. Privacy concerns can push Finland, Sweden and other Nordic countries towards redesigning their national identification numbers to omit the embedded personal information sometime in the future. There is an ongoing government project led by Ministry of Finance to redesign the system of personal identity codes (Valtiovarainministeriö, 2022). These and other related policy and legislation changes will be closely monitored and, if necessary, the package functions can be adjusted accordingly.

Both packages are published under permissive GPL-2 license. We encourage users to study the source code and modify it for their own use or submit improvements to our code repositories on GitHub.[10]

Acknowledgements

We are grateful to all contributors, in particular Juuso Parkkinen and Joona Lehtomäki for their support in the initial package development. This work is part of rOpenGov[11] LL was supported by Academy of Finland (decision 295741).

Bibliography

B. Ajana. *Governing through Biometrics: The Biopolitics of Identity*. Palgrave Macmillan, New York, 2013. [p1]

J. Brensinger and G. Eyal. The Sociology of Personal Identification. *Sociological Theory*, 2021. doi: 10.1177/07352751211055771. URL https://doi.org/10.1177/07352751211055771. OnlineFirst. [p1, 2, 6]

Digital and Population Data Services Agency. The personal identity code, 2022. URL https://dvv.fi/en/personal-identity-code. Accessed: 2022-01-17. [p6]

M. Foucault. *Security, territory, population: lectures at the Collège de France, 1977-1978*. Palgrave Macmillan, New York, 2009. Editors: Michel Senellart, François Ewald, Alessandro Fontana, Arnold I. Davidson. [p1]

W. Freitas. numbersBR: Validate, Compare and Format Identification Numbers from Brazil, 2018. URL https://CRAN.R-project.org/package=numbersBR. R package version 0.0.2. [p2]

P. Hendricks. generator: Generate data containing fake personally identifiable information, 2015. URL https://CRAN.R-project.org/package=generator. R package version 0.1.0. [p2]

T. Salste. Henkilötunnus – ihmisten koodaaja, 2021. URL https://www.tuomas.salste.net/doc/tunnus/henkilotunnus.html. Accessed: 2021-12-13. [p2]

Statistics Sweden. Personal identity number, 2016. [p2]

R. Sund. Quality of the Finnish Hospital Discharge Register: A systematic review. *Scandinavian journal of Public Health*, 40:505–15, 8 2012. doi: 10.1177/1403494812456637. [p2]

The Swedish Tax Agency. Personnummer: Skv 704 ed. 8, 2007. [p4]

Valtiovarainministeriö. Project on redesigning the system of personal identity codes, 2022. URL https://vm.fi/en/project-on-redesigning-the-system-of-personal-identity-codes. Accessed: 2022-01-17. [p6]

I. Watson. A short history of national identification numbering in Iceland. *Bifröst Journal of Social Science / Tímarit um félagsvísindi*, 1:51–89, 2010. ISSN 1670-7796. [p2]

[9] see Brensinger and Eyal (2021, 32) for a short description of synthetic fraud related to American SSNs

[10] https://github.com/rOpenGov/hetu, https://github.com/rOpenGov/sweidnumbr

[11] http://ropengov.org

H. Wickham and J. Bryan. R packages, 2022. URL https://r-pkgs.org/intro.html. The book is a work-in-progress. Accessed: 2022-01-17. [p2]

Åke Johansson. Från bläckpenna till datorhjärna. *Deklarationen 100 år och andra tillbakablickar*, 2003. [p2]

Pyry Kantanen
Department of Computing
PO Box 20014 University of Turku
Finland
ORCiD: 0000-0003-2853-2765
pyry.kantanen@utu.fi

Måns Magnusson
Department of Statistics
Uppsala University
Sweden
ORCiD: 0000-0002-0296-2719
mans.magnusson@statistik.uu.se

Leo Lahti
Department of Computing
PO Box 20014 University of Turku
Finland
ORCiD: 0000-0001-5537-637X
leo.lahti@iki.fi