Max-Planck-Institut für demografische Forschung

Max Planck Institute for Demographic Research

Konrad-Zuse-Strasse 1 ▪ D-18057 Rostock ▪ Germany ▪ Tel +49 (0) 3 81 20 81 - 0 ▪ Fax +49 (0) 3 81 20 81 - 202 ▪ www.demogr.mpg.de

MPIDR Technical Report TR 2018-001 I January 2018

R programs for splitting abridged fertility data into a fine grid of ages using the neural network method

Vasily P. Gorlishchev I gpa15@yandex.ru
Pavel Grigoriev I grigoriev@demogr.mpg.de
Anatoli I. Michalski I mpoctok@yandex.ru

For additional material see www.demogr.mpg.de/tr/

R programs for splitting abridged fertility data into a fine grid of ages using the neural network method

Vasily P. Gorlishchev (gpa15@yandex.ru)

Pavel Grigoriev (Grigoriev@demogr.mpg.de)

Anatoli I. Michalski (mpoctok@yandex.ru)

Abstract. The need to split aggregated fertility data into a fine grid of ages is a challenge that is often encountered by demographers. Several methods for addressing this problem have been developed. In this technical report, we present an application of a new approach to splitting abridged fertility data, the neural network (NN) model. Although neural networks have been widely used in various fields, they have seldom been applied in demography. The algorithm presented here is very flexible and simple to use, but it requires substantial computational resources. The NN method allows us to split abridged fertility rates of any kind using a pre-learned model generated with high-quality data drawn from the Human Fertility Database. The results of testing show that in most cases, the NN model returns estimates that correspond very closely to the original values. However, the model also tends to return erroneous estimates for fertility patterns that are 'unfamiliar' to the pre-learned model.

1.Background

The need to split aggregated fertility data into a fine grid of ages sometimes arises. To handle this task, several disaggregation methods have been developed and tested (McNeil et. 1975; Smith, Hyndman, Wood, 2004; Liu, et al. 2011; Schmertmann 2012; Jasilioniene et al. 2012, Grigoriev and Jdanov, 2015). Here, we present an application of the neural network (NN) method, which represents a new approach to splitting abridged fertility data. While neural networks have been widely used in various disciplines, they have seldom been applied in demography. To our knowledge, this is the first attempt to apply the neural network method to the problem of demographic data disaggregation. We intend to provide a *fine split* of aggregated fertility rates ($_nf_x$) into fertility rates by single years of age ($_1f_x$). We have established three criteria that must be met when applying the NN method in this context:

1) *Fit* – the predicted values should be as close as possible to the observed values;
2) *Shape* **-** the estimated fertility curves should be plausible and smooth; and
3) *Non-negativity* – the method should not generate negative values.

The NN method can be used to split any kind of input data with a structure that corresponds to a pre-learned model. If needed, a pre-learned model with the desired format can be easily generated by the user. Here, we describe a neural net that splits nine five-year age groups: 10-14, … , 50-54 into single ages from 12 to 54. The NN estimation algorithm contains a smoothing spline and negative value extinction procedures.

2.Description of the algorithm

Below, we provide a very brief and general description of the NN algorithm. A more detailed description of the process can be found elsewhere (Riedmiller and Braun, 1994). Our base model is a pre-learned neural network with a resilient back-propagation algorithm. The output of the network consists of smoothed estimates adjusted for negative values.

2.1. The neural network and the resilient back-propagation algorithm

The basic component of a neural net is a formal neuron, shown in Figure 1.

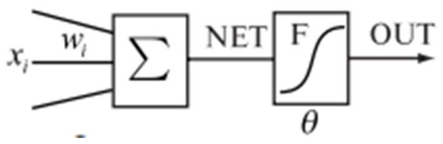


Figure 1. Formal neuron

Source: Adopted from Zaencev (1999)

A formal neuron contains a weighted sum and a non-linear element (activation function). The operation of a formal neuron can be described by the following formulas:

$$NET = \sum_i w_i x_i \quad (1)$$

$$OUT = F(NET - \theta), \quad (2)$$

where

x_i – the input signals, whereby the vector of all of the input values stands for x;

w_i – the weight coefficients;

NET – the weighted sum of all input values, whereby the *NET* value is transferred to the non-linear elements;

Θ – the threshold level of the neuron;

F – the non-linear function, which stands for the activation function – the sigmoidal (logistic) function that accepts and recalculates $(NET - \theta)$

$OUT = \frac{1}{1+e^{-NET}}$ - Output with a sigmoidal activation function, whereby θ equals zero for this function.

The combination of the formal neurons builds a neural net, shown in Figure 2.

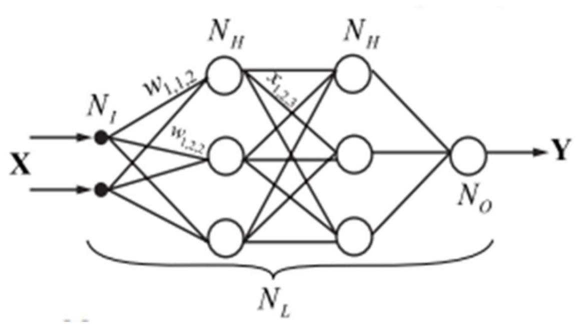


Figure 2. Formal structure of the neural network

Source: Adopted from Zaencev (1999)

Formally, the neural network is just a sequential evaluation of linear and non-linear function combinations:

$$f(x) = F\left(\sum_{i_k} w_{i_N j_N N} \ldots \sum_{i_2} w_{i_2 j_2 2} F\left(\sum_{i_k} wi_{1 j_1 1} - \theta_{j_1 1}\right) - \theta_{j_2 2} \ldots - \theta_{j_K N}\right) \qquad (3)$$

The sequential evaluation provides a close approximation of the multidimensional function.

The resilient back-propagation algorithm (2) is a method used to tune the weights *w*$_i$ in a way that allows the function *f*(*x*) to approximate data. After all of the train data have been entered into the neural net and all of the derivative errors of the formal neurons have been counted, the updated values $\Delta^{(t)}{}_{ij}$ for the neural net coefficients are calculated by the following system of equations:

$$\Delta^{(t)}{}_{ij} = \begin{cases} \eta^{+} * \Delta^{(t-1)}{}_{ij} & , \quad if \ \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} * \dfrac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ \eta^{-} * \Delta^{(t-1)}{}_{ij} & , \quad if \ \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} * \dfrac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ \Delta^{(t-1)}{}_{ij} & , \quad else \end{cases} \qquad (4)$$

where

η is a step parameter. The updated values of the weights are based on the information on a local error derivative

$$\Delta w^{(t)}{}_{ij} = \begin{cases} -\Delta^{(t)}{}_{ij} & , \quad if \ \dfrac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ +\Delta^{(t)}{}_{ij} & , \quad if \ \dfrac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ 0 & , \quad else \end{cases} \qquad (5)$$

The weights are updated by the following rule:

$$w^{(t+1)}_{ij} = w^{(t)}_{ij} + \Delta w^{(t)}_{ij} \tag{6}$$

The cycle continues until convergence is reached.

2.2. Spline and elimination of negative values procedures

The smoothing spline estimates value $\hat{f}$ for the neural net estimates f, and is by definition a minimizer for the function:

$$\sum_{i=1}^{n} (Y_i - \hat{f}(x_i))^2 + \lambda \int_{x_1}^{x_n} \hat{f}''(x)^2 \, dx \longrightarrow \min_{\hat{f}} \, , \tag{7}$$

where $\lambda \geq 0$ is a smoothing parameter that trades the quality of the approximation and the smoothness of function (3).

The vector of the smoothed spline parameters is calculated by the following formula:

$$\hat{m} = (I + \lambda A)^{-1} Y \tag{8}$$

with the matrix

$$A = \int f_i''(x) \, f_j''(x) \, dx \tag{9}$$

The negative values are eliminated by the following simple substitution:

$$\hat{f}(x_i) = \begin{cases} \hat{f}(x_i) & , if \ \hat{f}(x_i) \geq 0 \\ 0 & if \ \hat{f}(x_i) < 0 \end{cases} \tag{10}$$

2.3. Disaggregation of the neural network structure

We constructed a neural net with nine input neurons (age groups: 10-14, 15-19, … , 50-54), two hidden layers with 27 and 36 neurons, and 43 output neurons (single age groups: 12,13,14,…,54). The structure of this neural net appears in Figure 3.

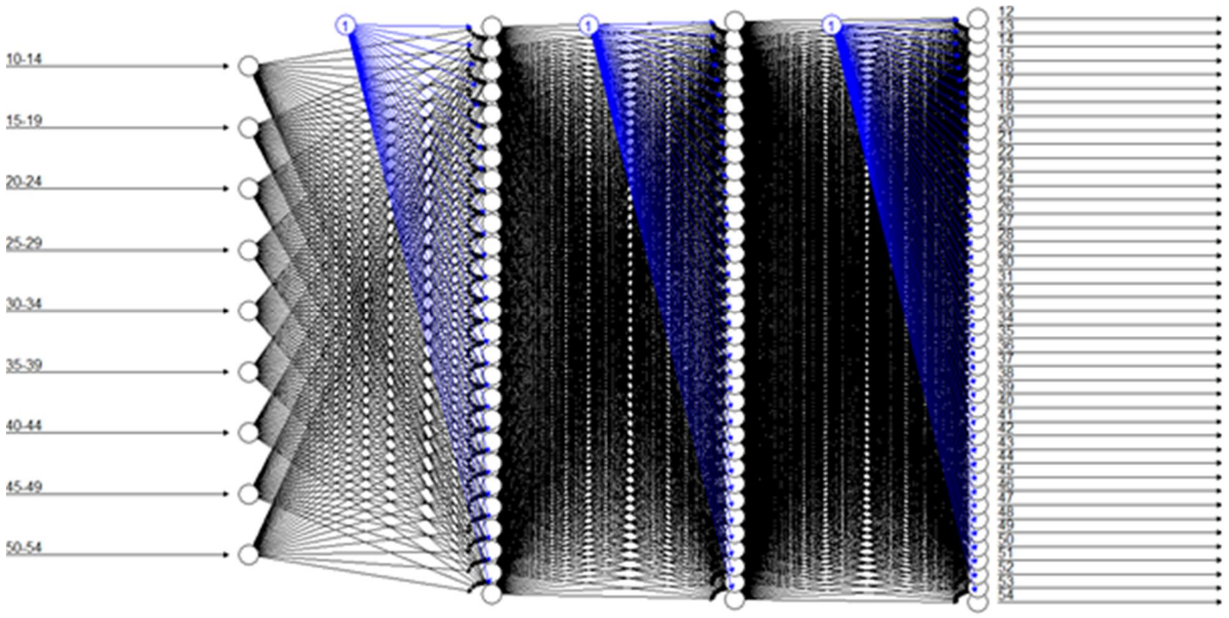


Figure 3. Structure of the neural net

3. Requirements

The scripts were tested using R version 3.1. The library ‘*neuralnet’* is required for the calculation of the neural network, and the library ‘bigsplines’ is required for the smoothing procedure. Both packages can be installed from the standard R repository (CRAN).

4.Usage

The .zip file included in the technical report contains *R scripts,* and the data files used in the example files *HFD_BirthsExp1x1.csv and HFC_ASFR_TOT 9 age groups.txt* were drawn from the HFD and the HFC, respectively. All of the other data files were derived on the basis of *HFD_BirthsExp1x1.csv,* and can be reproduced using the provided R scripts *‘1 Generating abridged data.R’ and* ‘2 Preparing data for learning.R’. The file *NN Functions.R* contains four functions that needed to be loaded into the R environment using the *source* command. A description of these functions is provided below.

4.1. R functions

After sourcing ‘*NN Functions.R’* (command ‘source’), the following R functions are ready for use: *NN.learn(), NN.split (), NN.plot(), and NN.errors().*

4.1.1. NN.learn()

This function builds a neural network on the basis of real observations (HFD data). The model establishes the associations between the disaggregated and the aggregated fertility rates generated from the same data source.

Usage: *NN.learn(input = train.input, output = train.output)*

The arguments are as follows:

train.input – the data frame (or the vector) containing the abridged age-specific fertility rates;

train.output – the data frame (or the vector) containing the disaggregated age-specific fertility rates (Explanatory note: As both *train.input* and *train.output* are generated from the original birth counts and the population exposures by single years of age, they have the same attributes as *CountryYear*. The format of these data is described in section 4.2.); and

layers – the vector of the hidden layers and the units in them, default: *layers=c(10,50,50,50,60).*

The accuracy of the neural network prediction is very sensitive to this parameter. How the *layers* are designed depends on both the quantity and the quality of the raw data. A larger amount of data generally implies a higher degree of heterogeneity. If there are 300+ examples (empirical fertility schedules), we recommend that the sizes of the layers be kept as small as possible: *c(10,50,50,50,60).* If the data are homogeneous and the number of schedules is small (about 30), the sizes of the layers can be larger: *c(10,50,50,50,50,50,50,50,50,60).*(Explanatory note: layers = c(20,20,20) – three-layer network with 20 units in each layer).

Example: *NN.learn(input = train.input, output = train.output)*

NN.learn returns a large R object that has to be passed into the *NN.split* function.

4.1.2. NN.split ()

This function splits abridged data (rates) into a fine grid of ages.

Usage: *NN.split (input = x, model = model1)*

The arguments are as follows:

x – the data frame (or the vector) containing the abridged age-specific fertility rates (Explanatory note: The format of the input data is described in section 4.2.); and

model1 – the pre-learned model.

Example: *NN.split (test.input[c('AUT1964','CAN1980'),], model = model1)*

NN.split returns the data frame containing the disaggregated age-specific fertility rates in the following format:

columns – ages by single years of age (12, 13, 14, ….,52, 53, 54)

rows – CountryYear (e.g., AUT1964)

Example of the output (transposed):

```
      AUT1965    CAN1980
12 0.00009951 0.00000000
13 0.00026307 0.00000000
14 0.00174004 0.00071945
15 0.00689436 0.00352892
16 0.02445333 0.01340019
17 0.05074924 0.02413369
18 0.08643147 0.03749711
19 0.12101206 0.05710087
20 0.14054130 0.06995949
21 0.15697429 0.08215975
22 0.16779605 0.09841259
23 0.17106072 0.10896298
24 0.17328453 0.12336094
25 0.17019851 0.13036878
26 0.16540037 0.13335822
27 0.15640507 0.12952442
28 0.14632630 0.12168024
29 0.13557170 0.11143938
30 0.12107552 0.09362149
31 0.11063618 0.08110425
32 0.09772666 0.06357454
33 0.08708050 0.05250953
34 0.07721435 0.04264718
35 0.06670614 0.03116163
36 0.05947780 0.02490329
37 0.04788512 0.01730801
38 0.04116032 0.01322619
39 0.03398012 0.00967506
40 0.02530419 0.00559102
41 0.01984139 0.00372593
42 0.01455716 0.00223163
43 0.00955392 0.00070228
44 0.00633050 0.00014468
45 0.00288936 0.00000000
46 0.00233072 0.00000000
47 0.00109580 0.00000000
48 0.00013841 0.00000336
49 0.00000000 0.00000000
50 0.00005253 0.00000278
51 0.00000000 0.00000826
52 0.00023370 0.00001688
53 0.00000000 0.00000000
54 0.00009951 0.00000000
```

4.1.3. NN.plot ()

This function visualizes the results of splitting using the NN method.

Usage: *NN.plot (input, input.ages, output.ages, prediction, output=FALSE, additional=FALSE, file.name='NN.plot output.pdf')*

The arguments are as follows:

input – the data frame with the abridged fertility data;

input.ages – the mid-point values of the abridged age intervals;

output.ages – the desired output ages;

prediction – the predicted values returned by *NN.split;*

output (optional) – the data frame containing the original data by single years of age (Explanatory note: Normally, these data are not available, but this option might still be used to evaluate the performance of the NN model), default: *output=FALSE*;

additional (optional) – the data frame containing the predicted values obtained from an alternative splitting method (e.g., HFD method, calibrated spline method, etc.), default: *additional=FALSE;* and

file.name (optional) – the name of the output .pdf file, default: *file.name='NN.plot output.pdf'.*

Example:

```
> NN.plot(input=test.input, input.ages=seq(12.5,52.5,5),
prediction=prediction1, output= test.output, output.ages=(12:54),
file.name = 'NN split 9 age groups Example 2.pdf')
```

NN.plot returns a .pdf file with plots showing the results of splitting.

4.1.4. NN.errors()

This *function* calculates the root mean squared error (RSME) between the observed and the predicted values by the following formula:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}}, \qquad (11)$$

where y_i and $\hat{y}_i$ - observed and predicted values, respectively. n - number of observations.

NN.errors is used to evaluate the performance of the NN method (for testing purposes).

Usage: *NN.errors<- function(output=test.output, prediction=prediction1)*

The arguments are as follows:

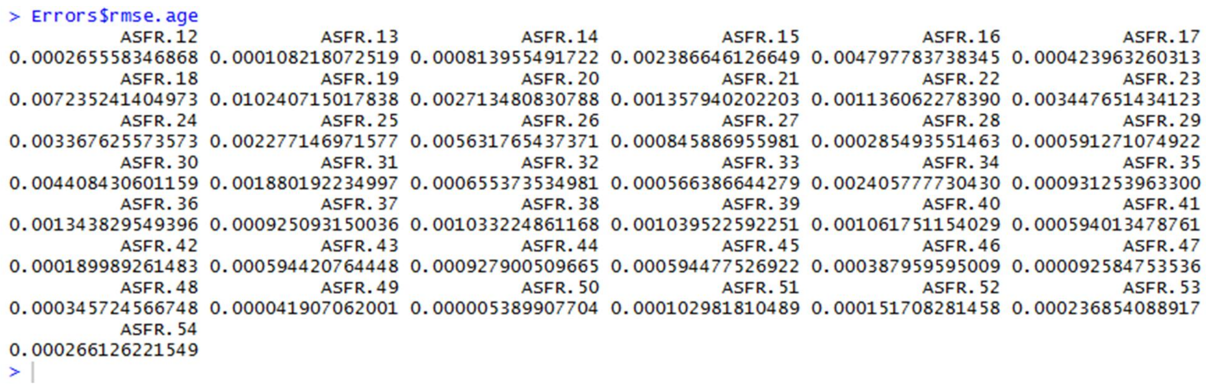
prediction – the data frame (or the vector) containing the predicted values of the ASFRs; and

output – the data frame (or the vector) containing the observed (real) values of the ASFRs.

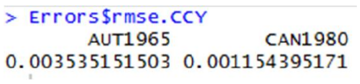
NN.errors returns a list containing *rmse.CCY* and *rmse.age* objects summarizing the RSME statistic by *CountryYear* or age*,* respectively.

Example:

```
> Errors<-NN.errors(output = test.output[c('AUT1965','CAN1980'),],
 prediction=prediction1[c('AUT1965','CAN1980'),])
```



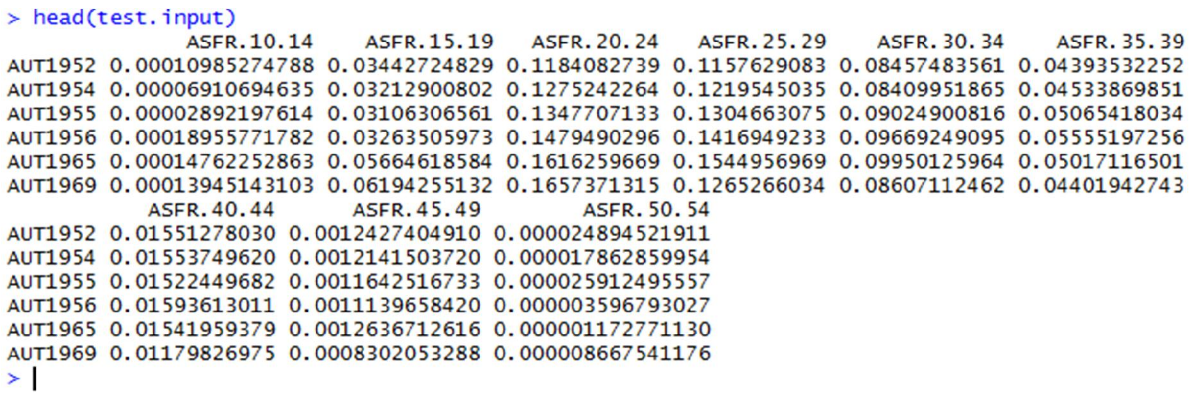
```
> Errors$rmse.age
           ASFR.12            ASFR.13            ASFR.14            ASFR.15            ASFR.16            ASFR.17 
0.000265558346868 0.000108218072519 0.000813955491722 0.002386646126649 0.004797783738345 0.000423963260313 
           ASFR.18            ASFR.19            ASFR.20            ASFR.21            ASFR.22            ASFR.23 
0.007235241404973 0.010240715017838 0.002713480830788 0.001357940202203 0.001136062278390 0.003447651434123 
           ASFR.24            ASFR.25            ASFR.26            ASFR.27            ASFR.28            ASFR.29 
0.003367625573573 0.002277146971577 0.005631765437371 0.000845886955981 0.000285493551463 0.000591271074922 
           ASFR.30            ASFR.31            ASFR.32            ASFR.33            ASFR.34            ASFR.35 
0.004408430601159 0.001880192234997 0.000655373534981 0.000566386644279 0.002405777730430 0.000931253963300 
           ASFR.36            ASFR.37            ASFR.38            ASFR.39            ASFR.40            ASFR.41 
0.001343829549396 0.000925093150036 0.001033224861168 0.001039522592251 0.001061751154029 0.000594013478761 
           ASFR.42            ASFR.43            ASFR.44            ASFR.45            ASFR.46            ASFR.47 
0.000189989261483 0.000594420764448 0.000927900509665 0.000594477526922 0.000387959595009 0.000092584753536 
           ASFR.48            ASFR.49            ASFR.50            ASFR.51            ASFR.52            ASFR.53 
0.000345724566748 0.000041907062001 0.000005389907704 0.000102981810489 0.000151708281458 0.000236854088917 
           ASFR.54 
0.000266126221549 
> |
```



```
> Errors$rmse.CCY
       AUT1965        CAN1980 
0.003535151503 0.001154395171 
```

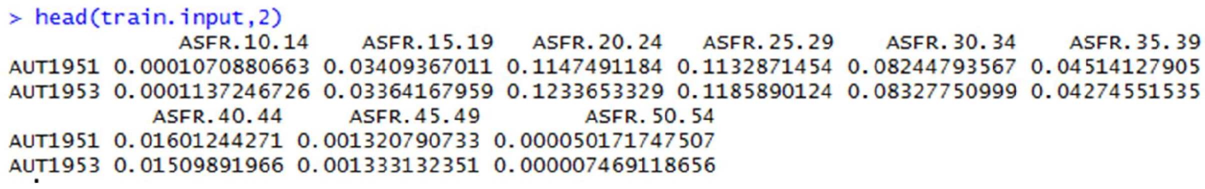
4.2 Format of the input data

The input should be either a data frame or a vector containing the age-specific fertility rates. A fragment of the aggregated input data to be disaggregated appears below:

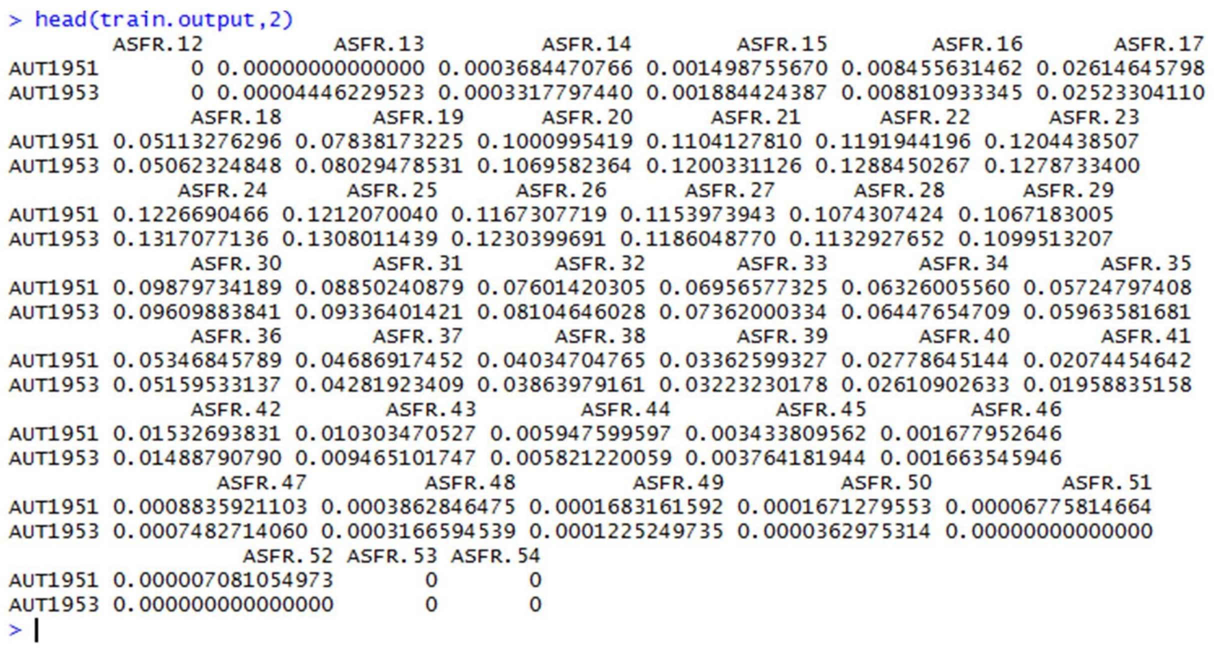


```
> head(test.input)
                ASFR.10.14     ASFR.15.19    ASFR.20.24    ASFR.25.29     ASFR.30.34     ASFR.35.39
AUT1952 0.00010985274788 0.03442724829 0.1184082739 0.1157629083 0.08457483561 0.04393532252
AUT1954 0.00006910694635 0.03212900802 0.1275242264 0.1219545035 0.08409951865 0.04533869851
AUT1955 0.00002892197614 0.03106306561 0.1347707133 0.1304663075 0.09024900816 0.05065418034
AUT1956 0.00018955771782 0.03263505973 0.1479490296 0.1416949233 0.09669249095 0.05555197256
AUT1965 0.00014762252863 0.05664618584 0.1616259669 0.1544956969 0.09950125964 0.05017116501
AUT1969 0.00013945143103 0.06194255132 0.1657371315 0.1265266034 0.08607112462 0.04401942743
           ASFR.40.44      ASFR.45.49        ASFR.50.54
AUT1952 0.01551278030 0.001242740491O 0.000024894521911
AUT1954 0.01553749620 0.001214150372O 0.000017862859954
AUT1955 0.01522449682 0.001164251673З 0.000025912495557
AUT1956 0.01593613011 0.0011139658420 0.000003596793027
AUT1965 0.01541959379 0.0012636712616 0.000001172771130
AUT1969 0.01179826975 0.0008302053288 0.000008667541176
> |
```

The data used for learning have the same format:



```
> head(train.input,2)
                ASFR.10.14     ASFR.15.19    ASFR.20.24    ASFR.25.29     ASFR.30.34     ASFR.35.39
AUT1951 0.0001070880663 0.03409367011 0.1147491184 0.1132871454 0.08244793567 0.04514127905
AUT1953 0.0001137246726 0.03364167959 0.1233653329 0.1185890124 0.08327750999 0.04274551535
           ASFR.40.44      ASFR.45.49        ASFR.50.54
AUT1951 0.01601244271 0.001320790733 0.000050171747507
AUT1953 0.01509891966 0.001333132351 0.000007469118656
```



```
> head(train.output,2)
        ASFR.12          ASFR.13         ASFR.14        ASFR.15        ASFR.16       ASFR.17
AUT1951       0 0.00000000000000 0.0003684470766 0.001498755670 0.008455631462 0.02614645798
AUT1953       0 0.00004446229523 0.0003317797440 0.001884424387 0.008810933345 0.02523304110
              ASFR.18       ASFR.19      ASFR.20      ASFR.21      ASFR.22      ASFR.23
AUT1951 0.05113276296 0.07838173225 0.1000995419 0.1104127810 0.1191944196 0.1204438507
AUT1953 0.05062324848 0.08029478531 0.1069582364 0.1200331126 0.1288450267 0.1278733400
             ASFR.24      ASFR.25      ASFR.26      ASFR.27      ASFR.28      ASFR.29
AUT1951 0.1226690466 0.1212070040 0.1167307719 0.1153973943 0.1074307424 0.1067183005
AUT1953 0.1317077136 0.1308011439 0.1230399691 0.1186048770 0.1132927652 0.1099513207
              ASFR.30       ASFR.31       ASFR.32       ASFR.33       ASFR.34       ASFR.35
AUT1951 0.09879734189 0.08850240879 0.07601420305 0.06956577325 0.06326005560 0.05724797408
AUT1953 0.09609883841 0.09336401421 0.08104646028 0.07362000334 0.06447654709 0.05963581681
              ASFR.36       ASFR.37       ASFR.38       ASFR.39       ASFR.40       ASFR.41
AUT1951 0.05346845789 0.04686917452 0.04034704765 0.03362599327 0.02778645144 0.02074454642
AUT1953 0.05159533137 0.04281923409 0.03863979161 0.03223230178 0.02610902633 0.01958835158
              ASFR.42        ASFR.43        ASFR.44        ASFR.45        ASFR.46
AUT1951 0.01532693831 0.010303470527 0.005947599597 0.003433809562 0.001677952646
AUT1953 0.01488790790 0.009465101747 0.005821220059 0.003764181944 0.001663545946
                ASFR.47         ASFR.48         ASFR.49         ASFR.50          ASFR.51
AUT1951 0.0008835921103 0.0003862846475 0.0001683161592 0.0001671279553 0.00006775814664
AUT1953 0.0007482714060 0.0003166594539 0.0001225249735 0.0000362975314 0.00000000000000
                 ASFR.52 ASFR.53 ASFR.54
AUT1951 0.00007081054973       0       0
AUT1953 0.00000000000000       0       0
> |
```

Note: The row and the column names in *train.input* and *train.output* must be identical. The same requirement applies to *test.input* and *test.output.*

5. Examples

For the sake of simplicity and the convenience of the user, the procedure for splitting aggregated fertility data is divided into three modules:

1) generating the abridged data,
2) preparing the data for learning, and
3) modeling and predicting.

The corresponding R scripts that perform these functions are included in the .zip file. Module (1) allows users to generate abridged fertility data of any kind, which can then be used for learning. By default, the age-specific fertility rates for the nine age groups used in this report are calculated on the basis of the birth counts and the population exposures by single years of age (see file '*HFD_BirthsExp1x1.csv*'). Module 2 contains the utilities for reshaping the input data in the internal format. Finally, using Module 3, the users can build their own neural network model and split the aggregated data into single ages.

As the scripts with examples are accompanied by detailed comments, even inexperienced R users should find the NN method very simple to use.

References

Grigoriev P, Jdanov DA (2015). Splitting abridged fertility data using different interpolation methods: is there the optimal solution?. 80th Annual Meeting of the Population Association of America 2015 (Hilton San Diego Bayfront, San Diego, CA, 2015) http://www.humanfertility.org/Docs/paa/Grigoriev_Jdanov.pdf

Hastie TJ. and Tibshirani RJ (1990). Generalized Additive Models, Vol. 43 of Monographs on Statistics and Applied Probability, Chapman and Hall, London.

Jasilioniene A, Jdanov DA, Sobotka T, Andreev EM, Zeman K, Nash EJ, and Shkolnikov VM (with contributions of Goldstein J, Philipov D. and Rodriguez G) (2012). Methods Protocol for the Human Fertility Database. Available at: http://www.humanfertility.org

Liu Y, Gerland P, Spoorenberg T, Kantorova V, Andreev K (2011). Graduation methods to derive age-specific fertility rates from abridged data: a comparison of 10 methods using HFD data. Presentation at the First Human Fertility Database Symposium, Max Planck Institute for Demographic Research, Rostock, Nov 2011. http://www.humanfertility.org/Docs/Symposium/Liu-Gerland%20et%20al.pdf

McNeil DR, Trussell TJ, Turner JC (1977). Spline interpolation of demographic data. *Demography* 14(2): 245-252

Riedmiller M, Braun H (1993). A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In Proceedings of the IEEE International Conference on Neural Networks 1993 (ICNN 93).

Schmertmann C (2012). Calibrated spline estimation of detailed fertility schedules from abridged data. MPIDR Working Paper WP-2012-022. Rostock.

Smith L, Hyndman R, Wood S (2004). Spline interpolation for demographic variables: the monotonicity problem. *Journal of Population Research* 21 (1), pp. 95-97.

Zaencev (1999). Neural networks: main models. The manual for the course "Neural networks" [Neironnye seti: osnovnye modeli. Uchebnoe posobie k kursu “Neironnye seti”]. Voronezh, 76 p.