2014•2015
FACULTY OF SCIENCES
Master of Statistics

Master's thesis

Use of a multinomial additive hazards model to assess the disability burden using cross-sectional data

Supervisor :
Prof. dr. Geert MOLENBERGHS

Supervisor :
Prof. Dr. HERMAN VAN OYEN

Renata Tiene De Carvalho Yokota

Thesis presented in fulfillment of the requirements for the degree of Master of Statistics

Transnational University Limburg is a unique collaboration of two universities in two countries: the University of Hasselt and Maastricht University.

universiteit hasselt
KNOWLEDGE IN ACTION

Universiteit Hasselt | Campus Hasselt | Martelarenlaan 42 | BE-3500 Hasselt
Universiteit Hasselt | Campus Diepenbeek | Agoralaan Gebouw D | BE-3590 Diepenbeek

universiteit hasselt | Maastricht University

2014•2015
FACULTY OF SCIENCES
Master of Statistics

Master's thesis

Use of a multinomial additive hazards model to assess the disability burden using cross-sectional data

Supervisor :
Prof. dr. Geert MOLENBERGHS

Supervisor :
Prof. Dr. HERMAN VAN OYEN

Renata Tiene De Carvalho Yokota

Thesis presented in fulfillment of the requirements for the degree of Master of Statistics

universiteit hasselt | Maastricht University

Use of a multinomial additive hazards model to assess the disability burden using cross-sectional data

Renata Tiene de Carvalho Yokota

Promotors: Professor dr. Geert Molenberghs
Profesor dr. Herman Van Oyen

Master Thesis nominated to obtain the degree of Master of Statistics, specialization Epidemiology and Public Health

Hasselt University, 2015

Abstract

The global phenomenon of population ageing results in a larger number of elderly individuals living with disability. Despite the importance of investigating the disability burden due to its social and economic impacts, few methods have been proposed to address the impact of chronic diseases on disability using cross-sectional data, which are cheaper and less time consuming than longitudinal studies. The attribution method based on the additive hazards model for binary outcomes is an attractive option, as it enables the partition of the disability prevalence into additive contributions of chronic diseases, taking into account comorbidity. The link function used in the model imposes a constraint on the parameter space, which limits the use of the method, as it is not available in standard software. Currently, the software to fit the binomial additive hazards model is available in R, but since it was developed to non R-users, it has limited efficiency when bootstrap confidence intervals are requested. Additionally, the constraints on the parameter space are taken into account by including a penalty term in the likelihood function. In this study, we propose an extension of the binomial model to a multinomial response and the use of linear constrained optimization to estimate the disability rates using the R function "constrOptim". For illustration, we assess the contribution of chronic diseases to the disability prevalence using the data from the Belgian Health Interview Surveys of 2001, 2004, and, 2008, with the Global Activity Limitation Indicator (GALI) as the binary and three-category outcome. The use of the parallel option to obtain the bootstrap confidence interval for the binomial model speeded up the analysis compared to the original R code. The models proposed can be used by health professionals to provide information to assist policy-makers on the development of strategies to tackle the disability burden. Further research should focus on the extension of the multinomial model to ordinal multi-category outcomes.

Keywords: *Binomial likelihood, multinomial likelihood, additive hazards model, cross-sectional data, chronic diseases, disability burden*

Contents

1

Introduction

The increase in longevity accompanied by the growing prevalence of chronic diseases contributes to the disability burden observed worldwide[1]. Chronic diseases are among the main causes of disability[2], which affects the quality of life of the elderly and increases the health care use, resulting in higher social burden. Given the growing occurrence and impact of disability especially in the older population, mortality measures, such as life expectancy, are no longer sufficient to assess population health. The investigation of population health should take into account both mortality and morbidity. Health expectancies, i.e. the number of remaining years spent in a health state from a particular age assuming current rates of morbidity and mortality, are examples of such health metrics[3].

The overall mortality can be obtained from the death certificates and disability can be assessed in national surveys. However, this information is not sufficient, as the knowledge of the main causes of mortality and morbidity are important to define prevention, intervention, and treatment strategies. Thus, information on cause-specific mortality and disability are also required in the assessment of population health. The cause-specific mortality data can also be obtained from the death certificates, but the assessment of disability by cause is more challenging[4].

Several methods have been proposed to assess the causes of disability. Although longitudinal studies can be considered the gold standard, they are usually expensive and with restricted sample size. Thus, the use of cross-sectional data under certain assumptions has become a popular alternative to estimate the disability prevalence by cause[5]. Most of the existing methods using cross-sectional data are based on logistic regression, with focus on the effect of elimination of specific causes on disability. However, the results are affected by the order that a cause is removed, which can produce inconsistent results in the presence of comorbidity[6]. Furthermore, since these methods are based on a multiplicative model, they do not yield additive contributions of the causes[5].

Recently, Nusselder and Looman (2004)[4] proposed the attribution method to assess the contribution of chronic diseases to the disability prevalence using cross-sectional data. The method is based on a binomial additive hazards model, which allows the partition of the disability prevalence into additive contributions of chronic diseases in the presence of comorbidity. The implementation of the model is challenging, as it requires a constraint on the parameter space to provide probabilities that lie between 0 and 1. The attribution method has been used to assess the disability prevalence by cause in several countries, including the Netherlands[5], Belgium[6], Germany[7], and China[8]. The software to fit the model was developed in R[9] and is available upon request to the authors of the method[4]. In the original R code, the constraints are implemented by including a penalty term in the likelihood function when the probabilities are less or equal than 0.

The main objective of this study was to extend the binomial model to a multinomial

response. Also, we aimed at modifying the existing R software for the binomial model to improve efficiency in the calculation of the bootstrap confidence interval. This thesis is organized as follows: in section 2, the description of the data used to fit the models is presented. In section 3, the attribution method, including the binomial and multinomial additive hazards models, are defined. In section 4, the application of the models to the Belgian Health Interview Survey data is shown. The discussion is presented in section 5. Finally, the conclusions and further recommendations are presented in section 6. The R code is included in the appendix.

2 Data description

The pooled data from three Health Interview Surveys (HIS) conducted in Belgium − 2001, 2004, and 2008 − were used in this study. The HIS is a national household survey representative of the Belgian population, including approximately 10,000 individuals per year, selected based on multi-stage sampling with geographical stratification and clustering. The response rate varied from 61% (2001 and 2004) to 55% (2008). The sample included elderly individuals living in nursing homes and homes for the elderly and proxy interviews. The complex sample design was taken into account by the inclusion of sample weights in the analysis. A detailed description of the HIS methodology can be found elsewhere[10].

Since the disability prevalence was low in young individuals (<5%), this analysis was restricted to men aged 55 years or older. The outcomes, both binary and multinomial, were based on the global activity limitation indicator (GALI)[11], defined by the question:

For at least the past six months, to what extent have you been limited because of a health problem in activities people usually do?

0. *Not limited at all*

1. *Limited, but not severely*

2. *Severely limited*

For the binomial model, options 1 and 2 were combined to represent the disabled individuals and for the multinomial model, the three options of answer were used. In both models, the category "*0. Not limited at all*" was the reference category.

The covariates included in this analysis were age, categorized in 10-years age groups (55-64 years; 65-74 years; 75-84 years; ≥85 years) and five diseases: chronic respiratory diseases, diabetes, heart attack, stroke, and arthritis. The GALI was included in the self-administered questionnaire and the disease questions were included in the face-to-face questionnaire in the three HIS. After excluding individuals with missing data on the response and diseases ($N = 2810$), the sample size was $N = 4356$. The age distribution and the prevalence of the diseases are presented in Table 1.

Table 1. Number of individuals and percentage according to each covariate included in the model. Health Interview Survey, Belgium, 2001, 2004, and 2008.

Covariate	N	%
Age		
55−64 years	1721	39.8
65−74 years	1387	32.1
75−84 years	834	19.3
≥85 years	384	8.9
Diseases		
Chronic respiratory diseases	593	13.7
Diabetes	417	9.6
Heart attack	538	12.4
Stroke	81	1.9
Arthritis	1153	26.7

3 Methods

3.1 Attribution method

The attribution method was used to estimate the disability prevalence by cause using cross-sectional data. Similar to the cause-specific mortality data, in which one disease is assigned as underlying cause of death according to the death certificate, the method aims to attribute each disability case reported in a survey to a single cause, taking into account that individuals can have more than one disease (comorbidity) and that disability can be present in individuals without any of the diseases included in the study[4].

Even if an individual reports a disease in the survey, this is not necessarily the cause of the disability. The disability that is not associated with the diseases included in the analysis is labelled "background". Disability in individuals who did not report any disease is entirely attributed to background, while disability in individuals who reported diseases is partitioned among the diseases and background. The background is represented by the intercepts (one for each age group) in the model.

The main assumptions of the method are: (i) the distribution of disability by cause is entirely explained by diseases that are still present at the time of the survey and by the background; (ii) the cause-specific disability rates for each disease were proportionally equal in the time preceding the survey; (iii) individuals from the same age group are exposed to the same background rate; and (iv) the start of the time at risk for disability is the same for all causes.

Analogous to the mortality analysis in the presence of competing risks, in which under

the assumption of independence between causes of death, an exponential transformation is applied to the cumulative force of mortality to obtain the cause-specific probability of death[12,13,14], we used hazards rates to obtain the probability of being disabled by cause. Under the additive assumption of the rates, the total disability rate can be obtained by adding up the cause-specific disability rates[14].

The attribution method was initially based on the binomial additive hazards model, for binary outcomes. In the next sections, the binomial and the its extension to a multinomial response are presented.

3.2 **Binomial additive hazards model**

Let $(y_i, \boldsymbol{x_i})$ represent the data for each individual $i(i = 1, ..., n)$ of a cross-sectional study, where $\boldsymbol{x_i'} = (x_{i1}, ..., x_{id})$ is the vector of covariates included in the model and y_i is the vector of the binary response variable for each individual i, defined as shown below.

$$y_i = \begin{cases} 1, & \text{if individual } i \text{ is disabled} \\ 0, & \text{otherwise} \end{cases}$$

The covariates in the model can be continuous, dichotomous or non-linear combinations of them. The binomial additive hazards model is defined in (1).

$$\begin{aligned} &Y_i \sim Bernoulli(\pi_i) \\ &\pi_i = 1 - \exp(-\eta_i) \\ &\eta_i = \boldsymbol{x_i'}\boldsymbol{\beta} \end{aligned} \tag{1}$$

To obtain the parameter estimates in model (1), the binomial log-likelihood function shown in (2) is maximized.

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{n} y_i \log(\pi_i) + (1 - y_i) \log(1 - \pi_i) \tag{2}$$

The covariance matrix of the parameter estimates was estimated based on the inverse of the observed information matrix, defined in (3).

$$-\frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_h \partial \beta_j} = \sum_{i=1}^{n} x_{ij} x_{ik} \frac{y_i(1 - \pi_i)}{(\pi_i)^2} \tag{3}$$

The regression coefficients in (1) represent the disability hazards rates for each covariate in the model, i.e. the the rate at which each covariate causes disability among the individuals with the covariate present (usually, chronic diseases and age) for categorical variables. For continuous covariates, the regression coefficients represent the change in the disability rate for one-unit increase in the covariate. In order to assess the contribution of each covariate to the disability prevalence, we calculate the probability of being disabled due to cause d, as shown in (4).

$$P(y_i = 1|x_{id}) = \frac{x_{id}\beta_d}{\eta_i} \times \pi_i \tag{4}$$

Next, we estimate the number of disabled individuals by each cause according to (5).

$$N_d = \sum_{i=1}^{n} P(y_i = 1|x_{id}) \tag{5}$$

Finally, the prevalence of disability by cause can be calculated as shown in (6):

$$\text{Prev}_d = \frac{N_d}{N} \tag{6}$$

where N is total number of individuals in the sample.

3.3 **Multinomial additive hazards model**

In the multinomial version of the additive hazards model, since the response variable y_i can have more than two categories, indicator variables are created for the response. Let $\boldsymbol{y_{ij}} = (y_{i0}, y_{i1}, \ldots, y_{iJ})$ denote the vector of responses for individual i for each j category of the response, defined as shown below:

$$y_{ij} = \begin{cases} 1, & \text{if } y_i = j \\ 0, & \text{otherwise} \end{cases}$$

with $\sum_{j=0}^{J} y_{ij} = 1$.

Model (1) is now extended to multinomial responses as shown in (7).

$$\begin{aligned} Y_{ij} &\sim Multinomial(n_i, \pi_{ij}) \\ \pi_{ij} &= 1 - \exp(-\eta_{ij}) \\ \eta_{ij} &= \boldsymbol{x_i'\beta_j} \end{aligned} \tag{7}$$

Where $\boldsymbol{x_i'} = (x_{i1}, ..., x_{id})$ is the vector of covariates included in the model and $\boldsymbol{\beta_j}$ is the vector of parameter estimates for each j category of the response. Since $\sum_{j=0}^{J} \pi_{ij} = 1$, results in one redundant response category (for example, consider the first category $j = 0$

as the reference), we can restrict j in (7) to be $j = 1, \ldots, J$, with $y_{i0} = 1 - \sum_{j=1}^{J} y_{ij}$ and $\pi_{i0} = 1 - \sum_{j=1}^{J} \pi_{ij}$.

The multinomial log-likelihood function is defined as shown in (8).

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{n} \{ \sum_{j=1}^{J} y_{ij} \log(\pi_{ij}) + (1 - \sum_{j=1}^{J} y_{ij}) \log(1 - \sum_{j=1}^{J} \pi_{ij}) \} \tag{8}$$

The covariance matrix can also be defined as the inverse of the information matrix, as in the binomial case. The information matrix now consists of $(J-1)^2$ blocks of size $d \times d$, and is defined as shown (9), for $j = j$:

$$-\frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_{jk} \partial \beta_{jk'}} = -\sum_{i=1}^{n} x_{ik} x_{ik'} \left\{ (1 - \sum_{j=1}^{J} y_{ij}) \left[\frac{(1-\pi_{ij})(1-\sum_{j=1}^{J} \pi_{ij}) - (1-\pi_{ij})^2}{(1-\sum_{j=1}^{J} \pi_{ij})^2} \right] + \sum_{j=1}^{J} y_{ij} \left[\frac{(\pi_{ij}-1)}{(\pi_{ij})^2} \right] \right\} \tag{9}$$

and for $j \neq j'$:

$$-\frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_{jk} \partial \beta_{j'k'}} = -\sum_{i=1}^{n} x_{ik} x_{ik'} \left\{ (1 - \sum_{j=1}^{J} y_{ij}) \left[\frac{(1-\pi_{ij})(1-\pi_{ij'})}{(1-\sum_{j=1}^{J} \pi_{ij})^2} \right] \right\} \tag{10}$$

The contribution of each covariate to the total disability prevalence can be obtained in the same way as in the binomial case, with equations (6), (7), and (8) applied to each $J-1$ category of the response, as shown below.

Probability of being disabled due to cause d for each category j of the outcome:

$$P(y_{ij} = 1 | x_{id}) = \frac{x_{id} \beta_{dj}}{\eta_{ij}} \times \pi_{ij} \tag{11}$$

Number of disabled individuals by each cause d in each j category of the outcome:

$$N_{dj} = \sum_{i=1}^{n} P(y_{ij} = 1 | x_{id}) \tag{12}$$

Prevalence of disability by cause d in each j category of the outcome:

$$\text{Prev}_{dj} = \frac{N_{dj}}{N} \tag{13}$$

3.4 Constrained optimization

One of the challenges when fitting the binomial and multinomial additive hazards models (1) and (7) is the constrained parameter space, which must satisfy the conditions, for the binomial model:

$$\boldsymbol{x_i'} \boldsymbol{\beta} > 0 \tag{14}$$

and for the multinomial case:

$$\boldsymbol{x}_i'\boldsymbol{\beta_j} > 0 \tag{15}$$

These constraints must be satisfied in order to produce probabilities (π_i and π_{ij}) that lie between 0 and 1. Therefore, if models (1) and (7) are fitted using the available software (for example, using the function "glm" in R), convergence problems may occur or wrong estimates may be provided. In this analysis, the likelihood functions (2) and (8) were maximized over the restricted parameter space using an adaptive barrier algorithm. The constrained optimization was implemented using the R function "constrOptim" in the "stats" package.

3.5 Bootstrap confidence intervals

The functions to fit the binomial and multinomial models give the user the option to choose between the Wald confidence interval (CI) or the bootstrap CI. When analysing cross-sectional data, weights can be used to take into account the complex sample design in the data analysis. In the presence of wide range of individual weights in the sample, the likelihood theory may not apply, and the use of standard errors based on (3), (9), and (10) may lead to wrong conclusions. In this case, the bootstrap CI can be used as an alternative to the Wald CI. In the function proposed, the bootstrap CI is based on the 2.5^{th} and 97.5^{th} percentiles of the bootstrap replicas. Additionally, bootstrap CIs are provided for the calculation of the contribution of the covariates to the disability prevalence. In both cases, the calculation of the bootstrap CIs is done by calling the R package "boots", which has the advantage of allowing parallel computations using all the cores of the computer, speeding up the computation. The bootstrap CIs presented in this study were based on 1000 replicas, using the parallel option with 4 cores in the Windows operating system.

4 Data analysis

In this section the binomial and multinomial models are applied to the HIS data in Belgium. Five chronic diseases and one intercept for each age group were included in the models. Also, since the prevalence of chronic diseases increases over age, the two-way interactions between age and disease were included in the models.

The attribution of disability to chronic diseases is a function of the disease-specific disability rate and the disease prevalence in the population. Table 2 shows the prevalence of chronic diseases in men by age group, taking into account the survey weights of each individual. Arthritis was the most common disease among men, followed by chronic respiratory diseases in men aged 55−84 years and heart attack in men aged 85 years or older.

Table 2. Disease prevalence and confidence intervals. Health Interview Survey, Belgium, 2001, 2004, and 2008.

Diseases	55−64 years	65−74 years	75−84 years	≥ 85 years
Chronic respiratory diseases	9.1 (7.5;10.7)	13.2 (11.1;15.5)	19.5 (16.0;23.1)	17.2 (10.4;25.2)
Diabetes	7.3 (5.8;9.0)	9.2 (7.3;10.9)	11.6 (8.7;14.7)	7.9 (4.7;11.7)
Heart attack	6.9 (5.4;8.6)	12.1 (10.0;14.2)	18.1 (14.4;22.0)	27.5 (19.5;35.8)
Stroke	0.6 (0.3;0.9)	1.4 (0.7;2.3)	2.9 (1.6;4.4)	3.6 (0.7;7.9)
Arthritis	20.6 (18.1;23.1)	28.6 (25.0;32.6)	29.0 (25.4;33.2)	36.6 (27.4;45.2)

4.1 **Binomial additive hazards model**

Table 3 shows the hazard rates, the Wald 95% CI and the bootstrap CI calculated with 1000 replicas. The Wald CI was obtained in 33 seconds while the boostrap CI was obtained in 2 hours. Despite the difference in the length of intervals, as expected, the conclusions were the same: stroke was not significant in men aged 55-64 and 65-74 years and diabetes was not significant for individuals aged 75-84 years. In oldest old men only chronic respiratory diseases were statistically significant. Heart attack was the most disabling disease in men aged 55−74 years, while stroke and chronic respiratory diseases were the most disabling diseases in men aged 75−84 years and 85 years or older, respectively.

Table 3. Disability hazards rates, Wald 95% confidence intervals (Wald CI), and bootstrap confidence intervals (Boot CI) for the binomial additive hazards model. Health Interview Survey, Belgium, 2001, 2004, and, 2008.

Cause	55−64 years			65−74 years			75−84 years			≥85 years		
	Est	Wald CI	Boot CI	Est	Wald CI	Boot CI	Est	Wald CI	Boot CI	Est	Wald CI	Boot CI
Background	0.15	0.13;0.17	0.12;0.18	0.19	0.16;0.23	0.15;0.24	0.39	0.32;0.46	0.30;0.50	0.54	0.42;0.67	0.35;0.82
CRD[1]	0.53	0.36;0.70	0.32;0.80	0.50	0.33;0.66	0.29;0.74	0.55	0.30;0.79	0.23;0.89	1.56	0.79;2.33	0.69;2.58
Diabetes	0.35	0.19;0.52	0.13;0.63	0.15	0.01;0.28	0.00;0.31	-0.02	-0.21;0.16	-0.21;0.26	-0.08	-0.40;0.24	-0.46;0.34
Heart attack	0.75	0.51;1.00	0.43;1.25	1.03	0.78;1.28	0.74;1.40	0.43	0.18;0.67	0.14;0.78	0.23	-0.08;0.53	-0.23;0.99
Stroke	0.53	-0.08;1.15	-0.15;1.24	0.12	-0.27;0.52	-0.21;0.84	1.46	0.44;2.47	0.48;3.45	0.58	-0.52;1.68	-0.55;7.31
Arthritis	0.31	0.21;0.40	0.20;0.43	0.19	0.10;0.27	0.07;0.34	0.48	0.31;0.66	0.25;0.73	0.16	-0.06;0.39	-0.21;0.64

[1] CRD: Chronic respiratory diseases

Table 4 shows the contribution of each chronic disease to the total disability prevalence. The bootstrap CI with 1000 replicas was obtained in 1 hour. As a result of the additivity of rates, the contribution of each covariate sum to the total disability prevalence. For all age groups, background was the main contributor to the disability prevalence, followed by arthritis in men aged 55-64 years and 75-84 years and by chronic respiratory diseases in men aged 65-74 years and the oldest old men. The total disability prevalence and the background increased over age group.

Table 4. Contribution of chronic diseases and background to the disability prevalence. Health Interview Survey, Belgium, 2001, 2004, and, 2008.

Cause	55−64 years	65−74 years	75−84 years	≥85 years
Background	13.02 (12.93;13.11)	15.96 (15.80;16.12)	28.00 (27.58;28.40)	36.66 (35.17;38.09)
Chronic respiratory diseases	3.19 (2.57;3.78)	4.28 (3.54;5.10)	6.10 (5.01;7.24)	10.56 (6.20;15.68)
Diabetes	1.84 (1.42;2.28)	0.97 (0.78;1.17)	-0.21 (-0.27;-0.15)	-0.45 (-0.70;-0.26)
Heart attack	3.13 (2.41;3.86)	6.65 (5.40;7.93)	4.58 (3.63;5.61)	3.60 (2.48;4.88)
Stroke	0.19 (0.10;0.31)	0.12 (0.06;0.20)	1.74 (0.91;2.79)	1.17 (0.21;2.48)
Arthritis	4.75 (4.20;5.31)	4.02 (3.46;4.57)	8.51 (7.43;9.72)	3.70 (2.80;4.59)
Total disability prevalence	26.12 (25.11;27.11)	31.99 (30.66;33.33)	48.72 (47.16;50.33)	55.24 (51.86;58.65)

4.2 **Multinomial additive hazards model**

Table 5 shows the results for the multinomial additive hazards model. The model with the Wald CI was obtained in 10.5 minutes. The most disabling diseases in individuals who reported limitations, but not severe, were heart attack for men aged 55−74 years and stroke in men aged 75−84 years. None of the diseases were significant in the oldest old men. The most disabling diseases for the severely disabled men were chronic respiratory diseases in the young and oldest old individuals, heart attack in men aged 65−74 years and stroke in men aged 75−84 years.

Table 5. Disability hazards rates and Wald 95% confidence intervals for the multinomial additive hazards model. Health Interview Survey, Belgium, 2001, 2004, and, 2008.

Cause	55-64 years	65-74 years	75-84 years	≥85 years
Limited				
Background	0.06 (0.05;0.07)	0.08 (0.06;0.09)	0.19 (0.16;0.22)	0.57 (0.46;0.68)
CRD[1]	0.20 (0.14;0.27)	0.18 (0.12;0.24)	0.15 (0.07;0.24)	0.15 (-0.11;0.40)
Diabetes	0.12 (0.06;0.19)	0.12 (0.05;0.18)	-0.02 (-0.09;0.05)	0.03 (-0.22;0.27)
Heart attack	0.33 (0.23;0.43)	0.40 (0.30;0.49)	0.11 (0.03;0.20)	-0.10 (-0.24;0.03)
Stroke	0.17 (-0.07;0.41)	0.06 (-0.09;0.20)	0.45 (0.18;0.73)	0.00 (-0.32;0.32)
Arthritis	0.12 (0.09;0.16)	0.08 (0.05;0.12)	0.18 (0.11;0.25)	-0.03 (-0.16;0.10)
Severely limited				
Background	0.01 (0.01;0.01)	0.02 (0.02;0.02)	0.04 (0.03;0.04)	0.14 (0.11;0.17)
CRD[1]	0.11 (0.08;0.14)	0.11 (0.08;0.14)	0.11 (0.07;0.15)	0.31 (0.09;0.54)
Diabetes	0.06 (0.03;0.08)	0.03 (0.01;0.05)	0.05 (0.02;0.08)	0.03 (-0.09;0.15)
Heart attack	0.12 (0.08;0.15)	0.13 (0.10;0.17)	0.03 (0.01;0.05)	0.15 (0.05;0.25)
Stroke	0.11 (-0.01;0.24)	0.08 (-0.01;0.17)	0.16 (0.06;0.26)	0.09 (-0.24;0.42)
Arthritis	0.03 (0.02;0.04)	0.03 (0.02;0.05)	0.10 (0.07;0.12)	0.13 (0.06;0.21)

[1] CRD: Chronic respiratory diseases

A binomial additive hazards model was fitted to the $J-1$ levels of the three-category outcome, with the category "*0. Not limited at all*" as the reference, for comparison with the results of the multinomial model in Table 5. Despite the difference in the parameter estimates, as expected[15], the overall conclusions were similar.

Table 6. Disability hazards rates and Wald 95% confidence intervals (CI) for the binomial additive hazards model. Health Interview Survey, Belgium, 2001, 2004, and, 2008.

Cause	55−64 years	65−74 years	75−84 years	≥85 years
Limited				
Age	0.13 (0.10;0.15)	0.17 (0.13;0.20)	0.31 (0.25;0.38)	0.35 (0.25;0.44)
Chronic respiratory diseases	0.37 (0.22;0.52)	0.31 (0.18;0.44)	0.39 (0.17;0.60)	1.42 (0.52;2.32)
Diabetes	0.29 (0.12;0.46)	0.12 (0.01;0.23)	-0.05 (-0.19;0.09)	-0.01 (-0.28;0.26)
Heart attack	0.59 (0.38;0.81)	0.91 (0.65;1.17)	0.34 (0.13;0.56)	0.16 (-0.07;0.39)
Stroke	0.40 (-0.32;1.12)	-0.04 (-0.28;0.20)	1.18 (0.26;2.10)	0.01 (-0.53;0.55)
Arthritis	0.27 (0.19;0.36)	0.09 (0.03;0.16)	0.40 (0.23;0.56)	0.17 (-0.02;0.36)
Severely limited				
Age	0.03 (0.02;0.04)	0.03 (0.02;0.04)	0.10 (0.06;0.14)	0.25 (0.14;0.37)
Chronic respiratory diseases	0.28 (0.15;0.40)	0.31 (0.18;0.45)	0.39 (0.18;0.61)	0.93 (0.36;1.50)
Diabetes	0.11 (0.01;0.20)	0.04 (-0.02;0.09)	0.05 (-0.07;0.17)	-0.11 (-0.30;0.08)
Heart attack	0.37 (0.17;0.57)	0.44 (0.25;0.63)	0.15 (-0.02;0.33)	0.14 (-0.10;0.38)
Stroke	0.22 (-0.24;0.69)	0.18 (-0.14;0.50)	0.92 (-0.07;1.92)	0.73 (-0.47;1.94)
Arthritis	0.06 (0.02;0.10)	0.14 (0.07;0.20)	0.23 (0.11;0.35)	0.06 (-0.10;0.22)

Table 7 shows the contribution of chronic diseases to the disability prevalence, for each outcome category. For both outcome categories the background was the main contributor to the disability burden across all age groups. For men who reported limitation, but not severe, the second disease in the rank was arthritis, for men aged 55-64 years and 75-84 years, heart attack was the main contributor in men aged 65-74 years and chronic respiratory diseases were the main contributor in the oldest old men. In severely limited individuals, chronic respiratory diseases contributed most to the disability burden in men aged 55-74 years and arthritis was the main contributor in men aged 75 years or older. The prevalence of non-severe limitations was higher than the prevalence of severe disability in all age groups.

Table 7. Contribution of chronic diseases and background to the disability prevalence, according to levels of the outcome. Health Interview Survey, Belgium, 2001, 2004, and, 2008.

5 Discussion

In this project, the existing R software to fit the binomial additive hazards model and to calculate the attribution of chronic diseases to disability was modified, resulting in more flexible and efficient code. Additionally, the R software for the multinomial additive hazards model was developed.

The main challenge in implementing the binomial and multinomial additive hazards model is the constraint on the parameter space. This constraint is required due to the link function ($1-\exp(-\eta_i)$ and $1-\exp(-\eta_{ij})$) used in the models: for η_i and $\eta_{ij} \leq 0$, the probability of

Cause	55−64 years	65−74 years	75−84 years	≥85 years
Limited				
Background	5.78	6.93	16.54	43.64
Chronic respiratory diseases	1.33	1.66	2.06	1.65
Diabetes	0.71	0.92	-0.20	0.19
Heart attack	1.50	3.18	1.51	-1.91
Stroke	0.07	0.06	0.77	0.00
Arthritis	2.18	1.95	3.93	-0.84
Total disability prevalence	11.57	14.70	24.60	42.74
Severely limited				
Background	1.09	1.98	3.49	12.32
Chronic respiratory diseases	0.71	1.12	1.71	2.41
Diabetes	0.32	0.22	0.61	0.22
Heart attack	0.51	0.87	0.45	3.38
Stroke	0.05	0.10	0.27	0.33
Arthritis	0.53	0.89	2.17	3.68
Total disability prevalence	3.22	5.17	8.69	22.34

being disabled, π_i and π_{ij} can be zero or negative. In the existing software for the binomial model, the constraint is included by adding a penalty term = 0.0001 when $\eta_i \leq 0$. In our software, the linear inequality constraint was included in the optimization routine.

The original R code to estimate the attribution of disability to chronic diseases using the binomial model also allows the estimation of the bootstrap CI for the attribution and the parameter estimates. However, the original software does offer the parallel option, drastically decreasing efficiency: the results for the same binomial model fitted in tables 3 and 4 were obtained in 8 hours. In this case, the use of the "boot" package in R, which already has the parallel option built in the R function "boot", reduced the duration of the analysis by approximately 4 times, when 4 cpus were used and 1000 bootstrap replicas were requested.

It is important to mention that the attribution method presented in this study has some limitations. Disability can be incorrectly attributed to chronic diseases in cases that disability onset precedes disease onset. Also, the use of survey data usually rely on self-reports of disability and chronic diseases, which are not as accurate as medical examination. Moreover, the background contribution can be overestimated when important causes of disability are not included in the analysis. For instance in our example, dementia and injuries, important disability causes, were not included in the analysis, as it was not systematically available in the three HIS.

Also, the software developed has some limitations. The use of the R function "constrOptim" for both binomial and multinomial models requires that the initial values provided by the user are in a feasible region, i.e. on the parameter space. Therefore, the user can have difficulties in defining the initial values for the models. Furthermore, the calculation of the bootstrap CI for the parameter estimates and attribution for the multinomial model is very time consuming. Due to this time limitation the bootstrap CIs for the multinomial model are not presented in this report. Therefore, alternative methods to calculate the confidence intervals for the regression coefficients, such as based on the Bayesian inference, should be considered. Finally, it is important to keep in mind that the software proposed

was only applied to one data set. In order to assess the behaviour of the models in different scenarios, it is important to perform simulations and application to other data.

6 Conclusion

In conclusion, the use of linear constrained optimization seems to perform well when fitting the binomial and multinomial additive hazards models, provided that the initial values lie inside the parameter space. No convergence problems were encountered when applying the method to HIS data in Belgium. However, it is important to keep in mind that the computation of the bootstrap CI for the multinomial model can be very time consuming, depending on the size of the data and the number of parameters included in the model. The functions presented in the appendix can be used to assess the disability prevalence using cross-sectional data and the results can subsidize policy-makes to tackle the disability burden. Further research can focus on performing simulations to assess the behaviour of the models in different scenarios, extending the multinomial model to ordinal multi-category responses, and using Bayesian inference to estimate the regression coefficients and standard errors in the models, using prior distributions to constraint the parameter space.

References

[1] Christensen, K., Doblhammer, G., Rau, R., Vauupel, J. W. (2009). Ageing populations: the challenges ahead. Lancet, 374(9696): 1198−1208.

[2] Verbrugge, L. M., Lepkowski, J. M., Imanaka, Y. (1989). Comorbidity and its impact on disability. The Milbank quarterly, 67(3−4): 450−484.

[3] Robine, J. M., Romieu, I., Cambois, E. (1999). Health expectancy indicators. Bulletin of the World Health Organization, 77(2): 181−185.

[4] Nusselder, W. J., Looman, C. W. (2004). Decomposition of differences in health expectancy by cause. Demography, 41(2): 315−334.

[5] Klijs, B., Nusselder, W. J., Looman, C. W., Mackenbach, J. P. (2011). Contribution of chronic disease to the burden of disability. PLoS One, 6(9): e25325.

[6] Yokota, R. T., Berger, N., Nusselder, W. J., Robine, J. M., Tafforeau, J., Deboosere, P., Van Oyen, H. (2015). Contribution of chronic diseases to the disability burden in a population 15 years and older, Belgium, 1997-2008. BMC Public Health, 15: 229.

[7] Strobl, R., Müller, M., Emeny, R., Peters, A., Grill, E. (2013). Distribution and determinants of functioning and disability in aged adults–results from the German KORA-Age study. BMC Public Health, 13: 137.

[8] Chen, H., Wang, H., Crimmins, E. M., Chen, G., Huang, C., Zheng, X. (2014). The contributions of diseases to disability burden among the elderly population in China. Journal of Aging and Health, 26(2): 261−282.

[9] R Core Team. (2014). R. A language and environment for statistical computing. Version[3.0.3]. Vienna, Austria: R Foundation for Statistical Computing.

[10] Demarest, S., Van der Heyden, J., Charafeddine, R., Drieskens, S., Gisle, L., Tafforeau, J. (2013). Methodological basics and evolution of the Belgian health interview survey 1997-2008. Archives of Public Health, 71(1): 24.

[11] Robine, J. M., Jagger, C., Euro-REVES Group. (2003). Creating a coherent set of indicators to monitor health across Europe: the Euro-REVES 2 project. European Journal of Public Health, 13(3): 6−14.

[12] Chiang, C. L. (1961) On the probability of death from specific causes in the presence of competing risks. Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, 4: 169−180.

[13] Clayton, D., Hills, M. (1993). Statistical Models in Epidemiology. Oxford, New York, Tokyo: Oxford University Press.

[14] Manton, K., Stallard, E. (1984). Recent Trends in Mortality Analysis. Orlando: Academic Press Inc.

[15] Agresti, A. (2002). Categorical data analysis. 2nd edition. New Jersey: John Wiley & Sons, Inc.

7

Appendix

7.1 R code

```
# BINOMIAL ADDITIVE HAZARDS MODEL ----------------------------------------------------

BinAddHazEst <- function(param, data, y, x, WGT = 1, std.weight = FALSE,
                      Boots = FALSE, NBoots = 0, paral = FALSE,
                      typepar = "snow", ncpus = 4){

  if (is.data.frame(x)){
    x <- as.matrix(x, nrow = nrow(x), ncol = ncol(x))
  }

  BinAddHazLogLik <- function(param, data, y, x, WGT = 1, std.weight = FALSE){
      if((sum(is.na(y)) > 0) | (sum(is.na(x)) > 0)) {
      stop("Missing values (NA) are not allowed")}

    wgt <- NULL
    if (is.null(WGT)){
    wgt = rep(1, nrow(data))}

    if (any(WGT != 1) & std.weight == TRUE){
      wgt = WGT/mean(WGT)}

    if (any(WGT != 1) & std.weight == FALSE){
      wgt = WGT}

    beta <- param[1: length(param)]

    eta_i <- as.vector(x %*% beta)
    pi_i <- 1 - exp(-eta_i)

    dev.resid <- function(y, pi_i, wgt){
      2 * wgt * (y * log(ifelse(y == 0, 1, y/pi_i)) + (1 - y) *
                   log(ifelse(y == 1, 1, (1 - y)/(1 - pi_i))))}

    LL <- sum(dev.resid(y, pi_i, wgt))
    return(LL)}


  BinLL <- constrOptim(theta = param, f = BinAddHazLogLik, ui = x, ci = rep(0, nrow(x)),
                           method = "Nelder-Mead", data = data, y = y, x = x,
                           std.weight = std.weight, WGT = WGT)

  # Wald CI
  if(Boots == FALSE){
      CovAddHaz <- function(data, param, x, y){
```

```
      eta_i <- as.vector(x %*% param)
      pi_i <- 1 - exp(-eta_i)

      WA <- diag(drop(-y*(1 - pi_i)/(pi_i)^2))

      V <- -1 * (t(x) %*% WA %*% x)
       return(V)
    }

    Vcov <- CovAddHaz(data = data, param = BinLL$par, x = x, y = y)
    InvCov <- solve(Vcov)
    StdError <- as.vector(sqrt(diag(InvCov)))

    CILow <- BinLL$par - (1.96 * StdError)
    CIHigh <- BinLL$par + (1.96 * StdError)
    pvalue <- round(2 * pnorm(-abs(BinLL$par/StdError)), 4)

    Results <- list(coefficients = BinLL$par, ResidualDeviance = BinLL$value,
                    df = nrow(data) - length(BinLL$par), CILow = CILow, CIHigh = CIHigh,
                    pvalue = pvalue, StdError = StdError, Vcov = Vcov)} else {

# Bootstrap CI
      require(boot)

      mystat <- function(data, indices) {
        m <-  constrOptim(theta = BinLL$par, f = BinAddHazLogLik, ui = x[indices,], ci = 0,
                          method = "Nelder-Mead", data = data[indices,], y = y[indices],
                          x = x[indices,],
                          std.weight = std.weight,
                          WGT = WGT[indices])
        return(m$par)
      }

      set.seed(224)
      if (paral == TRUE){
      BootResult <- boot(data = data, statistic = mystat, R = NBoots, parallel = typepar,
                         ncpus = ncpus)
      } else {
        BootResult <- boot(data = data, statistic = mystat, R = NBoots)
      }

      BootCI <- matrix(NA, ncol = 2, nrow = length(param))

      for (i in 1:length(param)){
        BootCI[i,] <- boot.ci(BootResult,  conf = 0.95, type = "perc", index = i)[[4]][, 4:5]
        colnames(BootCI) <- c("CILow", "CIHigh")
        rownames(BootCI) <- names(BinLL$par)
      }

      Results <- list(coefficients = BinLL$par, CILow = BootCI[, "CILow"],
```

```
            CIHigh = BootCI[, "CIHigh"],
            df = nrow(data) - length(BinLL$par),
            ResidualDeviance = BinLL$value)
    }
      return(Results)
}

# ATTRIBUTION - BINOMIAL ADDITIVE HAZARDS MODEL ----------------------------------------

AttBinAddHaz <- function(param, BinAddHaz.coef, attvar.coef, data, y, x.mat, attvar,
                         WGT = NULL, std.weight = FALSE, attrib = "abs", NBoots = 1000,
                         paral = TRUE, typepar = "snow", ncpus = 4){

  if (is.data.frame(x.mat)){
    x.mat <- as.matrix(x.mat, nrow = nrow(x.mat), ncol = ncol(x.mat))}

  BinAddHazLogLik <- function(param, data, y, x.mat, WGT = 1, std.weight = FALSE){
    if((sum(is.na(y)) > 0) | (sum(is.na(x.mat)) > 0)) {
      stop("Missing values (NA) are not allowed")}

    wgt <- NULL
    if (is.null(WGT)){
      wgt = rep(1, nrow(data))}

    if (any(WGT != 1) & std.weight == TRUE){
      wgt = WGT/mean(WGT)}

    if (any(WGT != 1) & std.weight == FALSE){
      wgt = WGT}

    beta <- param[1: length(param)]

    eta_i <- as.vector(x.mat %*% beta)
    pi_i <- 1 - exp(-eta_i)

    dev.resid <- function(y, pi_i, wgt){
      2 * wgt * (y * log(ifelse(y == 0, 1, y/pi_i)) + (1 - y) *
                   log(ifelse(y == 1, 1, (1 - y)/(1 - pi_i))))}

    LL <- sum(dev.resid(y, pi_i, wgt))
    return(LL)}

    attvar.mat <- model.matrix(~attvar - 1)
    x <- cbind(attvar.mat, x.mat)

  BinAddHazAtt <- function(param, data, y, x, WGT = 1, std.weight = FALSE){

    BinLL <- constrOptim(theta = param, f = BinAddHazLogLik, ui = x, ci = rep(0, nrow(x)),
      method = "Nelder-Mead", data = data, y = y, x = x, std.weight
      std.weight, WGT = WGT)
```

```
  return(BinLL$par)
}

Attrib <- function(BinAddHaz.coef, attvar.coef, data, y, x.mat, attvar, WGT = NULL,
                   std.weight = FALSE, attrib = "abs"){

  if((sum(is.na(y)) > 0) | (sum(is.na(x)) > 0)) {
    stop("Missing values (NA) are not allowed")
  }

  wgt <- NULL
  if (is.null(WGT)){
    wgt = rep(1, nrow(data))}

  if (any(WGT != 1) & std.weight == TRUE){
    wgt = WGT/mean(WGT)}

  if (any(WGT != 1) & std.weight == FALSE){
    wgt = WGT}

  if (is.factor(attvar)){
    attvar = attvar} else {
      attvar = factor(attvar)
    }

  haz.dis <- t(BinAddHaz.coef * t(x.mat))
  haz.back <- attvar.coef[attvar]

  eta_i <- haz.back + apply(haz.dis, 1, sum)
  pi_i <- 1 - exp(-eta_i)

  att.x.mat <- (haz.dis/eta_i) * pi_i
  att.back <- (haz.back/eta_i) * pi_i

  att.mat <- matrix(NA, nrow = (ncol(x.mat) + 3), ncol = nlevels(attvar))

  att.mat[1,] <-  tapply(wgt, attvar, sum)
  att.mat[2,] <-  tapply(pi_i * wgt, attvar, sum)
  att.mat[3,] <-  tapply(att.back * wgt, attvar, sum)

  for (i in 1: ncol(x.mat)){
    att.mat[3 + i,] <- tapply(att.x.mat[, i] * wgt, attvar, sum)
  }

  dimnames(att.mat) <- list(c("nn","disab","backgrnd", colnames(x.mat)), levels(attvar))

  attribution <- list()
  attribution2 <- list()
  att.final <- list()

  for(i in 1: ncol(att.mat)){
```

```
    attribution[[i]] <- att.mat[, i]
    row_sub <- attribution[[i]] != 0
    attribution2[[i]] <- attribution[[i]][row_sub]

    if(attrib=="rel"){
      att.final[[i]] <- attribution2[[i]][3: length(attribution2[[i]])]/
        attribution2[[i]][2]
    } else {
      att.final[[i]] <- attribution2[[i]][2: length(attribution2[[i]])]/
        attribution2[[i]][1]}
  }

  names(att.final) <- levels(attvar)
  final.list <- unlist(att.final)
  final.names <- vector()
  for(i in 1:length(att.final)){
    final.names <- c(final.names, paste0(names(att.final[[i]]), names(att.final)[i]))
  }
  Output <-  matrix(unlist(att.final), ncol =1,  byrow = TRUE,
                    dimnames = list(final.names, "Contribution"))
  return(Output)
}

# Bootstrap CI
require(boot)

mystat.att <- function(data, indices) {
  m1 <- BinAddHazAtt(param = c(abs(attvar.coef), abs(BinAddHaz.coef)),
      data = data[indices,], y = y[indices],
      x = cbind(model.matrix(~ attvar - 1), x.mat)[indices,],
      WGT = WGT[indices], std.weight = std.weight)

  m2 <- Attrib(BinAddHaz.coef = BinAddHaz.coef, attvar.coef = attvar.coef,
               data = data[indices,], y = y[indices], x.mat = x.mat[indices,],
               attvar = attvar[indices], WGT = WGT[indices], std.weight = std.weight,
               attrib = attrib)
  return(m2)
}

set.seed(224)

if (paral == TRUE){
BootResult <- boot(data = data, statistic = mystat.att, R = NBoots, parallel = typepar,
                   ncpus = ncpus)
} else {
  BootResult <- boot(data = data, statistic = mystat.att, R = NBoots)
}

BootCI <- matrix(NA, ncol = 2, nrow = nrow(BootResult$t0) * ncol(BootResult$t0))

for (i in 1:(nrow(BootCI))){
```

```
    BootCI[i,] <- boot.ci(BootResult,  conf = 0.95, type = "perc", index = i)[[4]][, 4:5]
    colnames(BootCI) <- c("CILow", "CIHigh")
    rownames(BootCI) <- rownames(Attrib(BinAddHaz.coef = BinAddHaz.coef,
                                        attvar.coef = attvar.coef,
                                        data = data, y = y, x.mat = x.mat, attvar = attvar,
                                        WGT = WGT, std.weight = std.weight, attrib = attrib))
  }

  Attribution <- as.vector(Attrib(BinAddHaz.coef = BinAddHaz.coef, attvar.coef = attvar.coef,
                                  data = data, y = y, x.mat = x.mat, attvar = attvar,
                                  WGT = WGT, std.weight = std.weight, attrib = attrib))
  AttResult <- cbind(BootResult$t0, BootCI)
  AttRes <- round(AttResult, 4)

  return(AttRes)
}

# MULTINOMIAL ADDITIVE HAZARDS MODEL ----------------------------------------------------

 MultAddHaz <- function(param, data, y, x, WGT = 1, std.weight = FALSE, Boots = FALSE,
  NBoots = 0, paral = FALSE, typepar = "snow", ncpus = 4){

  if (is.data.frame(x)){
    x <- as.matrix(x, nrow = nrow(x), ncol = ncol(x))}

  y.levels <- sort(unique(y))

  if(is.factor(y)){
    y. <- model.matrix(~ y - 1)}

  if(!is.factor(y)){
    y.. <- factor(y, levels = y.levels, labels = y.levels)
    y. <- model.matrix(~ y.. -1)}
  colnames(y.) <- y.levels
  y.resp <- y.[, -1]


 MultAddHazLogLik <- function(param, data, y, x, WGT = 1, std.weight = FALSE){

  if(any(is.na(y)) | (any(is.na(x)))) {
    stop("Missing values (NA) are not allowed")
  }

  wgt <- NULL
  if (is.null(WGT)){
  wgt = rep(1, nrow(data))}

  if (any(WGT != 1) & std.weight == TRUE){
    wgt = WGT/mean(WGT)}

  if (any(WGT != 1) & std.weight == FALSE){
```

```
   wgt = WGT}

 beta_ij <- matrix(unlist(split(param, cut(seq_along(param), ncol(y.resp),
  labels = FALSE))), ncol = ncol(y.resp))
 eta_ij <- matrix(NA, ncol = ncol(beta_ij), nrow = nrow(x))

 for (i in 1:ncol(beta_ij)){
  eta_ij[,i] <- as.vector(x %*% beta_ij[,i])
 }

 pi_ij <- apply(eta_ij, 2, function(x) {1 - exp(-x)})

 sum.y <- apply(y.resp, 1, sum)
 sum.pi_ij <- apply(pi_ij, 1, sum)

 LL <- -sum(2 * wgt * ((1 - sum.y) * log(ifelse(sum.y == 1, 1, (1 - sum.pi_ij)))) +
            apply((y.resp * log(ifelse(y.resp == 0, 1, (pi_ij)))), 1, sum))
   return(LL)}

require(Matrix)
sparse.mat <- paste0("bdiag(", paste0(rep("x,", ncol(y.resp)-1), collapse=""), "x)")
ui.const <- eval(parse(text = sparse.mat))
MultLL <- constrOptim(theta = param, f = MultAddHazLogLik, ui = ui.const,
                    ci = rep(0, nrow(ui.const)), control=list(maxit=1000),
                    method = "Nelder-Mead", data = data, y = y, x = x,
                    std.weight = std.weight, WGT = WGT)

Coeff <- matrix(unlist(split(MultLL$par, cut(seq_along(MultLL$par), ncol(y.resp)))),
  ncol = ncol(y.resp))
colnames(Coeff) <- colnames(y.resp)

# Wald CI
 if(Boots == FALSE){

  CovMultAddHaz <- function(param, x, y.resp){

     beta_ij <- param
     eta_ij <- matrix(NA, ncol = ncol(beta_ij), nrow = nrow(x))

     for (i in 1:ncol(beta_ij)){
       eta_ij[,i] <- as.vector(x %*% beta_ij[,i])
     }

     pi_ij <- apply(eta_ij, 2, function(x) {1 - exp(-x)})

     sum.y <- apply(y.resp, 1, sum)
     sum.pi_ij <- apply(pi_ij, 1, sum)

     # j = j
     jj.1 <- matrix(NA, nrow = nrow(x), ncol = ncol(y.resp))
     jj.2 <- matrix(NA, nrow = nrow(x), ncol = ncol(y.resp))
```

```
for (i in 1: ncol(y.resp)){
  jj.1[, i] <- (1 - sum.y) * ( (((1 - pi_ij[,i]) * (1 - sum.pi_ij)) -
    (1 - pi_ij[,i])^2)/(1 - sum.pi_ij)^2)
  jj.2[, i] <- sum.y * ((pi_ij[, i] - 1)/(pi_ij[, i])^2)
}
jjd <- jj.1 + jj.2

jj <- list()
for (i in 1:ncol(y.resp)){
  jj[[i]] <- t(x) %*% diag(jjd[,i]) %*% x
}

# j != j'
jj.prime <- list()

for (i in 1:ncol(y.resp)){
  jj.prime[[i]] <- (1 - sum.y) * ( ((1 - pi_ij) * (1 - pi_ij[, i]))/(1 - sum.pi_ij)^2)
}

jjp.mat <- matrix(unlist(jj.prime), ncol = ncol(y.resp)*ncol(y.resp))

seq <- NULL
for (i in 0:(ncol(y.resp) -1)){
  seq[i] = 1 + (i * ncol(y.))
}

jjp.mat2 <- jjp.mat[,-c(1,seq)]

if(ncol(y.resp) > 2) {
  dup.col <- duplicated(t(jjp.mat2))
  jjp.mat3 <- jjp.mat2[, !dup.col]
} else {
  jjp.mat3 <- jjp.mat2
}

jjp.d <- list()

for (i in 1:ncol(y.resp)){
  jjp.d[[i]] <- t(x) %*% diag(jjp.mat3[,i]) %*% x
}

if (ncol(y.resp) == 2){

  cov1 <- cbind(jj[[1]], jjp.d[[1]])
  cov2 <- cbind(jjp.d[[2]], jj[[2]])

  Vcov <- -1 * rbind(cov1, cov2)

} else {
```

```
    require(Matrix)
    cov.mat <- bdiag(jj)

    used <- 0

    a <- ncol(x)
    b <- ncol(y.resp)

    for (irow in 1:(b - 1)){

      tempMat <- jjp.d[[used + 1]]

      if ((used + 1) != length(jjp.d)){
        for (k in (used + 2):(used + b - irow)){
          tempMat <- cbind(tempMat, jjp.d[[k]])
        }
      }
      rows <- seq((irow - 1) * (a + 1), irow * a, length = a)
      columns <- seq(irow*a+1, a*b, length=a*b-irow*a)
      cov.mat[rows, columns] <- tempMat

      used <- used+b-irow
    }

    usedL <- 0

    for (irow in 2:b){

      tempMat <- jjp.d[[usedL+1]]

      if ((usedL+1) > 1){
        for (k in (usedL+2):(usedL+2-b+irow)){
          tempMat <- cbind(tempMat, jjp.d[[k]])
        }
      }
      rows <- seq((irow-1)*a+1, irow*a, length=a)
      columns <- seq(1, a*(irow-1), length=a*(irow-1))
      cov.mat[rows, columns] <- tempMat

      usedL <- usedL+2-b+irow
    }
    Vcov <- -1 * cov.mat
  }
    return(Vcov)}

Vcov <- CovMultAddHaz(param = Coeff, x = x, y.resp = y.resp)
InvCov <- solve(Vcov)
StdError <- as.vector(sqrt(diag(InvCov)))

Std <- matrix(unlist(split(StdError, cut(seq_along(StdError), ncol(y.resp)))),
 ncol = ncol(y.resp))
```

```
colnames(Std) <- colnames(y.resp)

CILow <- Coeff - (1.96 * StdError)
CIHigh <- Coeff + (1.96 * StdError)
pvalue <- round(2 * pnorm(-abs(Coeff/StdError)), 4)

colnames(CILow) <- colnames(y.resp)
colnames(CIHigh) <- colnames(y.resp)
colnames(pvalue) <- colnames(pvalue)

Results <- list(coefficients = Coeff, ResidualDeviance = MultLL$value,
df = nrow(data) - length(MultLL$par), CILow = CILow, CIHigh = CIHigh, pvalue = pvalue,
StdError = Std, Vcov = Vcov)} else {

# Bootstrap CI
require(boot)

mystat <- function(data, indices) {

  if (is.data.frame(x)){
    x <- as.matrix(x, nrow = nrow(x), ncol = ncol(x))}

  y.levels <- sort(unique(y))

  if(is.factor(y)){
    y. <- model.matrix(~ y - 1)}

  if(!is.factor(y)){
    y.. <- factor(y, levels = y.levels, labels = y.levels)
    y. <- model.matrix(~ y.. -1)}
  colnames(y.) <- y.levels
  y.resp <- y.[, -1]

  require(Matrix)
  sparse.mat <- paste0("bdiag(", rep("x[indices,],", ncol(y.resp)-1), "x[indices,])")
  ui.const2 <- eval(parse(text = sparse.mat))

  m <-  constrOptim(theta = rep(0.02, ncol(y.resp) * ncol(x)), f = MultAddHazLogLik,
                    ui = ui.const2, ci = rep(0, nrow(ui.const2)), control=list(maxit=1000),
                    method = "Nelder-Mead", data = data[indices,], y = y[indices],
                    x = x[indices,], std.weight = std.weight, WGT = WGT[indices])
  return(m$par)
}

set.seed(224)

  if (paral == TRUE){
    BootResult <- boot(data = data, statistic = mystat, R = NBoots, parallel = typepar,
                           ncpus = ncpus)
       } else {
         BootResult <- boot(data = data, statistic = mystat, R = NBoots)
```

```
        }

        BootCI <- matrix(NA, ncol = 2, nrow = length(param))

        for (i in 1:length(param)){
          BootCI[i,] <- boot.ci(BootResult,  conf = 0.95, type = "perc",
            index = i)[[4]][, 4:5]
          colnames(BootCI) <- c("CILow", "CIHigh")
          rownames(BootCI) <- names(MultLL$par)
        }

        Results <- list(coefficients = MultLL$par, CILow = BootCI[, "CILow"],
               CIHigh = BootCI[, "CIHigh"], df = nrow(data) - length(MultLL$par),
                        ResidualDeviance = MultLL$value)
        return(Results)
  }
}

# ATTRIBUTION - MULTINOMIAL ADDITIVE HAZARDS MODEL ----------------------------------------

AttMultAddHaz <- function(param, MultAddHaz.coef, attvar.coef, data, y, x.mat, attvar,
                          WGT = NULL, std.weight = FALSE, attrib = "abs", NBoots = 1000,
                          paral = TRUE, typepar = "snow", ncpus = 4){

  if (is.data.frame(x)){
    x <- as.matrix(x, nrow = nrow(x), ncol = ncol(x))
  }

  y.levels <- sort(unique(y))

  if(is.factor(y)){
    y. <- model.matrix(~y - 1)}

  if(!is.factor(y)){
    y.. <- factor(y, levels = y.levels, labels = y.levels)
    y. <- model.matrix(~y.. -1)
  }
  colnames(y.) <- y.levels
  y.resp <- y.[,-1]


  MultAddHazLogLik <- function(param, data, y, x, WGT = 1, std.weight = FALSE){

    if(any(is.na(y)) | (any(is.na(x)))) {
      stop("Missing values (NA) are not allowed")
    }

    wgt <- NULL
    if (is.null(WGT)){
      wgt = rep(1, nrow(data))}
```

```
  if (any(WGT != 1) & std.weight == TRUE){
    wgt = WGT/mean(WGT)}

  if (any(WGT != 1) & std.weight == FALSE){
    wgt = WGT}

  beta_ij <- matrix(unlist(split(param, cut(seq_along(param), ncol(y.resp), labels = FALSE))),
  ncol = ncol(y.resp))

  eta_ij <- matrix(NA, ncol = ncol(beta_ij), nrow = nrow(x))

  for (i in 1:ncol(beta_ij)){
    eta_ij[,i] <- as.vector(x %*% beta_ij[,i])
  }

  pi_ij <- apply(eta_ij, 2, function(x) {1 - exp(-x)})

  sum.y <- apply(y.resp, 1, sum)
  sum.pi_ij <- apply(pi_ij, 1, sum)

  LL <- -sum(2 * wgt * ((1 - sum.y) * log(ifelse(sum.y == 1, 1, (1 - sum.pi_ij)))) +
              apply((y.resp * log(ifelse(y.resp == 0, 1, (pi_ij)))), 1, sum))
  return(LL)}

  attvar.mat <- model.matrix(~attvar - 1)
  x <- cbind(attvar.mat, x.mat)

  require(Matrix)
  sparse.mat <- paste0("bdiag(", rep("x,", ncol(y.resp)-1), "x)")
  ui.const <- eval(parse(text = sparse.mat))

  MultLL <- constrOptim(theta = param, f = MultAddHazLogLik, ui = ui.const,
                        ci = rep(0, nrow(ui.const)), control=list(maxit=1000),
                        method = "Nelder-Mead", data = data, y = y,  x = x,
                        std.weight = std.weight, WGT = WGT)

  MultAddHazAtt <- function(param, data, y, x, WGT = 1, std.weight = FALSE){
   MultLL <- constrOptim(theta = param, f = MultAddHazLogLik, ui = ui.const,
                        ci = rep(0, nrow(ui.const)), control=list(maxit=1000),
                        method = "Nelder-Mead", data = data, y = y, x = x,
                        std.weight = std.weight, WGT = WGT)
    Estimate <- MultLL$par
    return(Estimate)
  }

Attrib <- function(MultAddHaz.coef, attvar.coef, data, y, x.mat, attvar, WGT = NULL,
                   std.weight = FALSE, attrib = "abs"){

  if(any(is.na(y)) | (any(is.na(x)))) {
    stop("Missing values (NA) are not allowed")
  }
```

```
wgt <- NULL
if (is.null(WGT)){
  wgt = rep(1, nrow(data))}

if (any(WGT != 1) & std.weight == TRUE){
  wgt = WGT/mean(WGT)}

if (any(WGT != 1) & std.weight == FALSE){
  wgt = WGT}

id.disab <- list()
hdis <- list()
haz.back <- list()
eta_ij <- list()
pi_ij <- list()
att.x.mat <- list()
att.back <- list()
att.mat <- list()

for (i in 1:ncol(y.resp)){
  id.disab[[i]] <- which(y == 0 | y == i)
  hdis[[i]] <- t(MultAddHaz.coef[, i] * t(x.mat))
  haz.back[[i]] <- attvar.coef[,i][attvar]
  eta_ij[[i]] <- haz.back[[i]] + apply(hdis[[i]], 1, sum)
  pi_ij[[i]] <- 1 - exp(-eta_ij[[i]])
  att.x.mat[[i]] <- (hdis[[i]]/eta_ij[[i]]) * pi_ij[[i]]
  att.back[[i]] <- (haz.back[[i]]/eta_ij[[i]]) * pi_ij[[i]]

  att.mat[[i]] <- matrix(NA, nrow = (ncol(x.mat) + 3), ncol = nlevels(attvar))
  att.mat[[i]][1,] <-  tapply(wgt[id.disab[[i]]], attvar[id.disab[[i]]], sum)
  att.mat[[i]][2,] <-  tapply(pi_ij[[i]][id.disab[[i]]] * wgt[id.disab[[i]]],
                              attvar[id.disab[[i]]], sum)
  att.mat[[i]][3,] <-  tapply(att.back[[i]][id.disab[[i]]] * wgt[id.disab[[i]]],
                              attvar[id.disab[[i]]], sum)
  for (j in 1: ncol(x.mat)){
    att.mat[[i]][3 + j,] <- tapply(att.x.mat[[i]][, j][id.disab[[i]]] * wgt[id.disab[[i]]],
        attvar[id.disab[[i]]], sum)
    }

  dimnames(att.mat[[i]]) <- list(c("nn","disab","backgrnd", colnames(x.mat)),
                            levels(attvar))
}

attribution <- rep( list(list()), ncol(y.resp))
attribution2 <- rep( list(list()), ncol(y.resp))
att.final <- rep( list(list()), ncol(y.resp))
names(att.final) <- colnames(y.resp)
final.list <- list()

for(i in 1: ncol(y.resp)){
```

```
      for(j in 1: ncol(att.mat[[1]])){
        attribution[[i]][[j]] <- att.mat[[i]][, j]
        row_sub <- attribution[[i]][[j]] != 0
        attribution2[[i]][[j]] <- attribution[[i]][[j]][row_sub]

        if(attrib=="rel"){
         att.final[[i]][[j]] <- attribution2[[i]][[j]][3: length(attribution2[[i]][[j]])]/
        attribution2[[i]][[j]][2]
        } else {
         att.final[[i]][[j]] <- attribution2[[i]][[j]][2: length(attribution2[[i]][[j]])]/
           attribution2[[i]][[j]][1]}
      }
      names(att.final[[i]]) <- levels(attvar)
      final.list[[i]] <- unlist(att.final[[i]])

    }

    final.list <- unlist(att.final)
    final.names <- vector()
    for(j in 1:ncol(att.mat[[1]])){
        final.names <- c(final.names, paste0(names(att.final[[1]][[j]]),
          names(att.final[[1]])[j]))
      }

    Output <-  matrix(unlist(att.final), ncol = ncol(y.resp),
                      dimnames = list(final.names, paste0("Contribution", 1:ncol(y.resp))))
    return(Output)

  }

# Bootstrap CI
  require(boot)

  mystat.att <- function(data, indices) {

    if (is.data.frame(x)){
      x <- as.matrix(x, nrow = nrow(x), ncol = ncol(x))
    }

    y.levels <- sort(unique(y))

    if(is.factor(y)){
      y. <- model.matrix(~y - 1)}

    if(!is.factor(y)){
      y.. <- factor(y, levels = y.levels, labels = y.levels)
      y. <- model.matrix(~y.. -1)
    }
    colnames(y.) <- y.levels
    y.resp <- y.[,-1]
```

```
    require(Matrix)
    sparse.mat <- paste0("bdiag(", rep("x[indices,],", ncol(y.resp)-1), "x[indices,])")
    ui.const2 <- eval(parse(text = sparse.mat))

    m1 <- constrOptim(theta = rep(0.02, ncol(y.resp) * ncol(x)), f = MultAddHazLogLik,
       ui = ui.const2, ci = rep(0, nrow(ui.const2)),
       control=list(maxit=1000), method = "Nelder-Mead",
       data = data[indices,], y = y[indices], x = x[indices,],
       std.weight = std.weight, WGT = WGT[indices])

    m2 <- Attrib(MultAddHaz.coef = MultAddHaz.coef, attvar.coef = attvar.coef,
                 data = data[indices,], y = y[indices], x.mat = x.mat[indices,],
                 attvar = attvar[indices], WGT = WGT[indices], std.weight = std.weight,
                 attrib = attrib)

    return(m2)
  }

  set.seed(224)

  if (paral == TRUE){
  BootResult <- boot(data = data, statistic = mystat.att, R = NBoots, parallel = typepar,
                     ncpus = ncpus)
  } else {
    BootResult <- boot(data = data, statistic = mystat.att, R = NBoots)
  }

  BootCI <- matrix(NA, ncol = 2, nrow = nrow(BootResult$t0) * ncol(BootResult$t0))

  for (i in 1:(nrow(BootCI))){
    BootCI[i,] <- boot.ci(BootResult,  conf = 0.95, type = "perc", index = i)[[4]][, 4:5]
    colnames(BootCI) <- c("CILow", "CIHigh")
    rownames(BootCI) <- rep(rownames(Attrib(MultAddHaz.coef = MultAddHaz.coef,
                                            attvar.coef = attvar.coef, data = data,
                                            y = y, x.mat = x.mat, attvar = attvar,
                                            WGT = WGT, std.weight = std.weight,
                                            attrib = attrib)), ncol(y.resp))
  }

  Attribution <- as.vector(Attrib(MultAddHaz.coef = MultAddHaz.coef, attvar.coef = attvar.coef,
                                  data = data, y = y, x.mat = x.mat, attvar = attvar, WGT = WGT,
                                  std.weight = std.weight, attrib = attrib))
  AttResult <- cbind(Attribution, BootCI)
  AttRes <- round(AttResult, 4) *100

  return(AttRes)
}
```

Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Use of a multinomial additive hazards model to assess the disability burden using cross-sectional data

Richting: **Master of Statistics-Epidemiology & Public Health Methodology**
Jaar: **2015**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

De Carvalho Yokota, Renata Tiene

Datum: **8/09/2015**