

Analysis and Graphical Representation of Health and Labour Force Participation among the Elderly in Europe

Humboldt-Universität zu Berlin
School of Business and Economics
Ladislaus von Bortkiewicz Chair of Statistics



Statistical Programming Languages
Winter 2017/18

Seminar Paper by

Claudia Günther, Phi Nguyen, Julian Winkel
576419, 526624, 562959

Berlin, 2018-03-15

List of Tables

List of Figures

Abbreviations

xx xxxxxxxxxxxx

Contents

1	Introduction	2
2	Panel data cleaning and subsetting	3
2.1	Theory and Design	3
2.2	Implementation	3
2.3	Empirical Results	3
3	Multidisciplinary and crossnational summary statistics	4
3.1	Theory and Design	4
3.2	Implementation	4
3.3	Empirical Results	4
4	Crosssectional probit regression	5
4.1	Theory and Design	5
4.2	Implementation	5
4.3	Empirical Results	5
5	Wald Test	6
5.1	Theory and Design	6
5.2	Implementation	6
5.3	Empirical Results	8
6	Counterfactual exercise	10
6.1	Theory and Design	10
6.2	Implementation	10
6.3	Empirical Results	12
7	Graphical representation	13
7.1	Theory and Design	13
7.2	Implementation	13
7.3	Empirical Results	13
8	Conclusion	14
9	References	15

1 Introduction

- relevance of exploring relationship between health and labour force participation due to changing demographic in Europe
- cite relevant study about ageing
- few sentences about relevant papers exploring relationship -> use paper from DIW
- very short literature overview on relationship between health and labour force participation
- share data set as rich data set for this purpose: 2 sentences about it
- introduction of journal article
- our approach: replicate results and enrich analysis
- especially: introduce graphical visualization tools for descriptive statistics -> ease interpretation of variables
- our aim: write code in a way that allows the user to work with easySHARE data set, even when working on different question

2 Panel data cleaning and subsetting

PHI

2.1 Theory and Design

2.2 Implementation

2.3 Empirical Results

The easySHARE dataset released in spring 2017 is a panel dataset of 108 variables of more than 100.000 individuals covering data from six survey waves carried out between 2004 and 2005. As we are only concerned with a small subset of observations, an important task was to define appropriate functions for subsetting. In order to make our subsetting process understandable to readers, we decided on using pipe-operator. This allows us to apply the filter and select option in a clever way, where we can select different criteria at once.

-> code snippet here

In this example, we first filter for participation in wave 1 and the age group between 50 and 64 and then select the desired variables as described in Kalwij and Vermeulen (2005).

Although the overall response rate in the SHARE is comparably high, the data set still has numerous missing values. The reason for this is due to the fact that the study was carried out on a crossnational scale, with some national survey institutions deciding not to participate in all survey modules. This means that the majority of missing values are to be found within observations that have missing values for entire survey modules or waves. The reason for the missing values are documented well in the “Guide to easySHARE release 6.0.0” and specifically coded. For example, the numbers -13 and -14 refer to “not asked in this wave” and “not asked in this country”. Since this coding scheme is not useful for the purpose of our analysis, we decided on recoding all of the missing values as “NA”. To this end, we defined a function based on the missing codes provided by SHARE that finds the NAs in the data and declares them as such.

-> code snippet here

Since the study carried out by Kalwij and Vermeulen (2005) is based on the use of mostly binary data, we needed to construct numerous dummies based on the original data.

-> code snippet here

The resulting dataframe contains XXX observations of YYY variables.

- coded countries in more readable manner
- use package dplyr -> match official ISO code with country name
- create country list with all countries in study (not Israel)
- defined dummies

3 Multidisciplinary and crossnational summary statistics

JULIAN

3.1 Theory and Design

3.2 Implementation

3.3 Empirical Results

4 Crosssectional probit regression

JULIAN

4.1 Theory and Design

4.2 Implementation

4.3 Empirical Results

5 Wald Test

In the context of probit regression, the Wald test can be used to test multiple hypothesis regarding the model specifications and significance of coefficients. For example, it can be used to test whether the fit of the model is improved if a subset of regression coefficients are all set equal to zero, an exclusion restriction. Kalwij and Vermeulen (2005) conduct a Wald test to check the null hypothesis that none of the included health variables has an impact on labour participation in order to investigate the joint impact of health on participation.

5.1 Theory and Design

Depending on the estimation method and distributional assumptions, the Wald statistic can be formulated in different ways. The general form of the Wald statistic after MLE for testing hypothesis regarding our $k \times 1$ parameter vector θ is given by

$$W = c(\hat{\theta})'[\nabla_{\theta}c(\hat{\theta})\hat{V}\nabla_{\theta}c(\hat{\theta})]^{-1}c(\hat{\theta}) \quad (***)$$

where $c(\hat{\theta})$ is a $m \times 1$ vector of linear or nonlinear restrictions, $\nabla_{\theta}c(\hat{\theta})$ is the $m \times k$ Jacobian of $c(\hat{\theta})$ evaluated at $\hat{\theta}$ and \hat{V} is the estimated asymptotic covariance matrix (Wooldridge 2010, 463). Under H_0 , the Wald test statistic is asymptotically χ^2_m distributed, with m being the number of specified restrictions. In order to assure the test statistic W has the assumed limiting distribution, we need to impose some practical restrictions. Under H_0 , θ must lie within parameter space and R must be of rank m (Wooldridge 2010, 362). We limit our attention to testing a set of general linear restrictions since the Wald test is not invariant to the re-formulation of non-linear hypothesis (Wooldridge 2010, 362). We thus formulate our nullhypothesis in accordance with the common linear restriction structure of $H_0: R\hat{\beta} = r$ againsts the alternative $H_1: R\hat{\beta} \neq r$ to facilitate the derivation of our test statistics where R is a $m \times k$ matrix of rank m (equivalent to the Jacobian), whereas the restriction function r is a $m \times 1$ vector. Given the above technical conditions are satisfied, the Wald statistic can then be rewritten as

$$W = (R\hat{\beta} - r)'[R\hat{V}R']^{-1}(R\hat{\beta} - r) \quad (***)$$

which facilitates our calculations (Greene 2012, 527–29; Wooldridge 2010, 362). When we are interested in the joint significance of a subset of s coefficients, the nullhypothesis is that each of the s coefficients in the vector β_s is equal to zero. The Wald test statistic can then be even more simplified:

$$W = \hat{\beta}_s'[R\hat{V}R']^{-1}\hat{\beta}_s = \hat{\beta}_s'[\hat{V}_s]^{-1}\hat{\beta}_s \quad (***)$$

where the test statistic is distributed as $W \stackrel{a}{\sim} \chi^2_s$ (Wooldridge 2010, 362).

5.2 Implementation

In order to test linear hypothesis regarding the model specifications and significance of coefficients we design two versions of a Wald test. The function `joint.wald.test` is a simplified version to test the joint significance of a subset of model coefficients, whereas `general.wald.test` allows checking any linear hypothesis. The two tests are constructed in a way so they align well with the general glm estimation output, especially relevant for probit regression. An advantage over alternative Wald test functions from other packages (e.g. `aod`) is that it allows for empty class membership, as discussed in detail below.

We designed the joint significance test that makes it both easy to use, understand and modify to special needs. The only required input is the model summary from glm estimation, whereas the significance level and test specifications are optional.

```
joint.wald.test = function(model.summary, signif.level = 0.95, spec = NULL){

  joint.wald.test      = numeric(6)
  names(joint.wald.test) = c("Name", "W", "p-value", "df", "H0", "Decision")
  beta                 = model.summary$coefficients[,1]
  Var_beta_est         = vcov(model.summary)
```

In order to set up default values for the hypothesis specification, we make use of a local if-else statement inside the `joint.wald.test` function. By declaring the model specification to be null in line XXX we set the foundation for the default specification in line XXX - XXX. Here, we re-assign a sequence vector to the `spec` that includes the number of all model coefficients in increasing order, unless `spec` is specified differently by the user. In that case, the if-else statement will use the user specification assured by line XXX.

```
spec = if (is.null(spec)){
  spec = 1: length(beta) # default is joint significance test
} else {
  spec = spec}
```

This means that `joint.wald.test` function will conduct a joint significance test on all model coefficients by default. We setup a default significance level of 95% in the same manner. The Wald test statistic is calculated via simple matrix algebra based on the provided input by the model summary. This formula is equivalent to equation YYY.

```
W = t(beta[spec]) %*% solve(Var_beta_est[spec,spec]) %*% beta[spec]
```

As can be seen in line XXX the proper format of the specification vector `spec` is crucial as it is used to extract the needed estimates from the model coefficient vector and covariance matrix. In line XXX, the term `Var_beta_est[spec,spec]` extracts all covariance matrix elements corresponding to the joint significance hypothesis of the particular specification. To assure the proper extraction of the covariance elements and determination of degrees of freedom in line XXX, `spec` must be a vector of integers of length $0 < m \leq k$.

```
chi2      = qchisq(signif.level, df = length(spec))
pval      = 1-pchisq(W,length(spec))
joint.wald.test[1] = "Chi2 test"
joint.wald.test[2] = format(W, digits = 4)
joint.wald.test[3] = format(pval, digits = 4)
joint.wald.test[4] = length(spec)
joint.wald.test[5] = "b equal to 0"
joint.wald.test[6] = ifelse(pval <= 1- signif.level, "Reject H0", "Cannot reject H0")
joint.wald.test
```

As a last step we set up the test output by assigning values to the empty vector elements of `joint.wald.test`. This vector will be the function output that is returned. We determine the critical value and p-value in lines

XXX-XXX based on the inbuilt `qchisq` and display four significant decimal places. As can be seen in lines XX, we do not only list the test statistic and p-value but also the degrees of freedom, nullhypothesis and test decision as output to assure comprehensibility and correctness of the test.

The general Wald test is designed in a similar manner, except that it allows for the general formulation of linear hypothesis of the form $R\hat{\beta} = r$. The test statistic in this case is giving by equation YYY. As in the `joint.wald.test` function, we use local local if-else statements to setup a default settings. If only the model summary is given as an input, the `general.wald.test` conducts a joint significance test at a 95% significance level. The crucial part to the proper usage of this function are the correctly specified Jacobian matrix R and restriction vector r, which must be of size $m \times k$ and $m \times 1$ respectively. To assure the proper asymptotic distribution of the test statistic we additional need to assures that R is of rank m.

We therefore incorporate ...

5.3 Empirical Results

We use our designed wald test functions to carry out several hypothesis test on our model coefficients from probit estimation. Specifcically, we check the null hypothesis that none of the four included health variables has an impact on labour participation for each country and for both men an women. The results for men can be seen in table XXX

Our results our in line with the findings of Kalwij and Vermeulen (2005). As expected, the null hypothesis of no impact of health on labour participation is rejected at a 95% level for most countries. A particularity is the high p-value of France, leading to the inability to reject H0, which stands in contrast to the findings of Kalwij and Vermeulen (2005). We explain this with our differing sample, which is almost 50% larger and the lower overall employmen rate of our sample. Regarding the Wald tests for women, our results are very similar to those Kalwij and Vermeulen (2005). As for men, The majority of null hypothesis can be rejected for women, with the null for German women being barely not rejected. The inability to reject the null for some countries is due to relatively low health-related probit coefficients (in absolute terms), which implies that these health indicators are not strongly related with the labor participation decision. The results of our `joint.wald.test` function are comparable those of our packages, like the `wald.test` from the aod packages. To check our function this we can conduct both test on all subsets. For example, for German women the aod packages gives the following output

```
wald.test(Sigma = vcov(allModels$Germany.FEMALE),
          b = allModels$Germany.FEMALE$coefficients, Terms = 16:19)

Wald test:
-----

Chi-squared test:
X2 = 9.2, df = 4, P(> X2) = 0.057
```

Equivalently, our `joint.wald.test` function return as output

```
joint.wald.test(allSummaries$Germany.FEMALE, spec = 16:19)
```

Test	W	p-value	df	H0	Decision
"Wald"	"9.17"	"0.0569"	"4"	"b equal to 0"	"Cannot reject H0"

A weakness of the aod package is, that its wald test cannot deal with empty classes. For the case of Switzerland, where our male sample does not include any 64-year-olds, the `wald.test` from the aod packages lead to the error message

```
wald.test(Sigma = vcov(allModels$Switzerland.MALE),
          b = allModels$Switzerland.MALE$coefficients, Terms = 16:19 )
> Error in L %*% V : non-conformable arguments
```

Our wald test function is robust against the presence of empty classes since we work with model summaries, meaning that the coefficients of empty classes have been dropped instead of being a missing value. Thus, our `joint.wald.test` function can conduct the test for Swiss men without errors:

```
joint.wald.test(allSummaries$Switzerland.MALE, spec = 16:19)
```

Test	W	p-value	df	H0	Decision
"Wald"	"9.91"	"0.042"	"4"	"b equal to 0"	"Reject H0"

Both our wald test functions are sensitive to the improper formulation of tested model restrictions. For example, the `spec` vector must be a numerical vector specifying the coefficients to be included given their ordering in the model summary. The test also works when `spec` is specified as a logical vector but will return an inappropriate test distribution:

```
specification = c(rep(FALSE,15), rep(TRUE, 4), rep(FALSE,3))

joint.wald.test(allSummaries$Switzerland.MALE, spec = specification)
```

Test	W	p-value	df	H0	Decision
"Wald"	"9.91"	"0.987"	"22"	"b equal to 0"	"Cannot reject H0"

As in line XXX, our specification and Wald statistic are similar. However, the function fasely assigns 22 degrees of freedoms since this is the length of our specification vector.

6 Counterfactual exercise

Counterfactual exercises are a common tool in economic policy analysis that allow assessing and quantifying possible effects of policies. In order to evaluate the quantitative importance of a healthy population on labour participation, Kalwij and Vermeulen (2005) conduct a counterfactual exercise. They compare the current national employment rate to the predicted counterfactual rate. This counterfactual employment rate is obtained by assuming that all individuals are in perfect health. According to their definition, a perfectly healthy individual has never had a severe or mild medical condition, suffers from no restrictions in activities of daily living, is not obese and has a grip strength of an average 50-51 year-old (fe)male individual.

6.1 Theory and Design

Counterfactual exercises based on regression methods that can be carried out in different ways (see Chernozhukov et al 2013). The simplest method to is to replace the existing covariates values X with counterfactual values X_{cf} . For the case of probit regression with a binary response variable y for employment, we can interpret the fitted values \hat{y} and \hat{y}_{cf} as current and counterfactual employment probabilities for each individual. Based on these calculated individual employment probabilities, we can calculate the current and counterfactual population employment rates (Π_{ac}, Π_{cf}) by taking the mean, making use of the Law of Large Numbers. Obtaining the counterfactual probabilities \hat{y}_{cf} requires a two-step procedure. First, the regular probit regression of y on X is run to obtain the estimated coefficients $\hat{\beta}$. Second, the counterfactual probabilities for each individuals Π_i are calculated by predicting y based on X_{cf} and the estimated coefficients.

$$\Pi_{icf} = Pr(Y_i = 1 | X_i = x_{icf}) = \Phi(x_{icf}'\beta) \quad (***)$$

We can then assess counterfactual health-related effects by comparing the actual and counterfactual employment rates Π_{ac} and Π_{cf} . We can gain closer insights into the nonlinear relation between health effects and participation by calculating Π_{ac} and Π_{cf} for different age groups. This also allows assessing the proportion of decline in participation that is due to decline in health condition according to our model estimates.

6.2 Implementation

Our counterfactual effects are based on the manipulation of explanatory variables according to some specified criteria. We design this counterfactual exercise in a way that allows us to directly use the output from our probit estimation. As a first step, we create a simple function `empl.rate` that takes an estimation model object as input and predicts the employment rate, i.e. the mean of fitted values, as an output. In order to calculate the both current and counterfactual employment rates Π_{ac} and Π_{cf} conveniently, we need to manipulate the original covariates contained in our model objects. For this purpose we design the function `X.cf`, whose only input the model object, allowing us to treat each country separately later on. As can be seen in line XXX, the function creates a duplicate version of original data `X`, whose values will be replaced later on.

In line XXX, we calculate the minimum value for each variable in `X`, which corresponds to the perfect health conditions of the variables `h_chronic`, `h_adlaTRUE`, and `h_obeseTRUE` since we are dealing with standardized variables. We obtain these values with making use of the `apply` function and store them in the vector `X_min`.

We also calculate the mean average grip strength of 50- to 51-years old in the sample `X`. As can be seen in line XXX, we derive this mean with the `tapply` command because it allows us to specify an age group index and returns a vector whose elements we can access.

```
X                = model$data
X_cf             = X


X_min            = data.frame(t(apply(X, 2, min)))

names.vec        = c("h_chronic", "h_adlaTRUE", "h_obeseTRUE")
X_cf[, names.vec] = X_min[names.vec]

age_50_51        = ifelse(X$age50 == 1 | X$age51 == 1, 1, 0)

X_cf[, c("h_maxgrip")] = tapply(X$h_maxgrip, age_50_51 == 1, mean)[[2]]

model$data       = X_cf
```

Listing 5: Quantlet 4 

To obtain the counterfactual covariates, the values in `X_cf` are replaced with perfect health values. As a final step, we replace the original model data in `X_cf`. This allows us to access both the model estimates and counterfactual covariates in one object. By running our `empl.rate` function on the returned model, we directly obtain the counterfactual employment rate for a given country.

Since we are also interested in the current and counterfactual employment rates of different age groups, we create a more advanced function `empl.rate.age`. This function displayed in line XXX requires the estimation model as mandatory argument and has two optional arguments (`group.size` and `group.low`) that allow to change the age group and group sizes. Given the specified arguments, the function then calculates the current and counterfactual employment rate for each age group.

```
1 empl.rate.age      = function(model, group.size = 5, group.low = c(50,55,60)){
```

The proper value assignment of the numerical vector `group.low` is essential since the dynamic creation of age groups is based on it. As can be seen in line XXX, we use a small for loop to determine the upper age group bound and create a label of each age group. Furthermore, we assign all of the corresponding age values (e.g. `age50`, `age51`) to the vector `z`, which correspond to the age variable names in `X`.

```
    for (i in group.low){
46         k = i + group.size - 1
        vec.label = paste("names.vec.age", i, sep=".")
        z = assign(vec.label, paste0("age", i:k))
```

We need to assign the age values to `z` in order to locate the right individuals in our data frame `X`. We find the individuals belonging to a given age group by summing over rows of `X` and storing this vector inside the list `IND_row_vec` displayed in line XXX. We can take the sum because the age values assigned to `z` correspond to the column names of the data frame `X`. Thus, for a given individual, the row sum of `X[, z]` is either zero or one, depending on whether the individual belongs to the age group or not. Following this step, we filter out

only the relevant age groups via indexing.

```
IND_row_vec[[i]] = apply(X[, z], 1, sum)
IND_row_vec      = IND_row_vec[group.low]
```

Finally, the employment rate for each age group is calculate in line XXX. For this, we first sum the predicted employment probability for all individuals in age group k via vector multiplication. Next, we devide this sum by the overall number of individuals in that age group, corresponding to the vector length `IND_row_vec[[k]]`.

```
empl.rate[k] = empl.probability %*% IND_row_vec[[k]] / sum(IND_row_vec[[k]])
```

Running the `empl.rate.age` function on a given model object will return a numerical vector containing the employment rate of the specified age groups.

6.3 Empirical Results

7 Graphical representation

PHI

7.1 Theory and Design

7.2 Implementation

7.3 Empirical Results

8 Conclusion

9 References

Greene, William H. 2012. *Econometric Analysis*. Seventh Edition. Pearson Education.

Kalwij, Adriaan, and Frederic Vermeulen. 2005. “Labour Force Participation of the Elderly in Europe: The Importance of Being Healthy.”

Wooldridge, Jeffrey M. 2010. *Econometric Analysis of Cross Section and Panel Data*. MIT press.

Declaration of Authorship

We hereby confirm that we have authored this Seminar paper independently and without use of others than the indicated sources. All passages which are literally or in general matter taken out of publications or other sources are marked as such.

Berlin, 2018-03-15

Claudia Günther, Phi Nguyen, Julian Winkel