Discussion Paper

Bootstrapping the SPAR index

Léon Willenborg and Sander Scholtus

November 29, 2018

This paper discusses bootstrapping for the SPAR index for the development of house prices. Software (in R) to compute the SPAR index, written by the department at CBS responsible for the statistics of house prices and commercial buildings, is used in the bootstrap computations. The main problem applying the bootstrap is to generate suitable samples (in the form of input files) that could be fed to this SPAR tool. How such a tool - called a bootstrap sampler for the SPAR index (BSS) - should work is discussed in some detail. Furthermore the results from BSS for each bootstrap sample and to compute quantities such as variances need to be collected and compiled. This is the task of the bootstrap collector for the SPAR index (BCS). The more interesting tool of this pair is the BSS and some space is devoted to its description. Also the bootstrap results from our computations are discussed. As the bootstrap computations had to be performed on rather sizable data sets, efficiency was an issue. However, the bootstrapping for this problem can be parallelized, at least the computational intensive part of the work. This option should be explored if bootstrapping the SPAR index is to be performed regularly in the future.

1 Introduction

In the present paper[1)] we discuss a specific bootstrap problem, namely that concerning the SPAR index (SPAR stands for Sale Price Appraisal Ratio). This index is used to quantify the price development of private as well as commercial property. In our discussion we concentrate on houses (private property), because we had received data for this type of property and not for the other.[2)]

Willenborg and Scholtus (2018) is a companion paper to the present document.[3)] This paper gives some background information of the SPAR index. For instance, it uses a simple idea to derive various house price indices, among them the SPAR index. The results produced by these indices when applied to the same data set can be compared, in particular to the SPAR index. Some of these alternative indices use imputation to get rid of certain missing values in the data. The current implementation of the SPAR index at CBS avoids the use of imputation. So it is interesting to see whether imputation matters a lot or not.

The remainder of the paper is structured as follows. In Section 2 the input data that is used to compute the SPAR index is described. For the bootstrapping we use the same input data. In fact to compute the SPAR index we use the R code that was developed by the department for house prices. Section 3 is about the bootstrapping that we applied. It briefly discusses bootstrapping in general, but it mainly focusses on the bootstrapping for the SPAR index. In particular it considers a problem about the controlling of the number of records with missings in bootstrap samples. In

1) The authors would like to thank Ron van Schie and Farley Ishaak (both from the Department of House Prices) for providing us with the data set and an R module to compute the SPAR index, and for their comments on an earlier version of this paper. We are also grateful to Frank Pijpers and Arnout van Delden (both of the Methodology Department), who reviewed mature versions of the present paper.

2) The main difference between both types of property is the number of transactions involved: many more houses are sold than commercial property. This makes it possible to compute the SPAR index for private property on a monthly basis, whereas for commercial property a quarterly basis was chosen; there are too few sales of commercial property for an index on a monthly basis.

3) In fact, it is a spin-off from the present paper. Its genesis is explained in the introduction of Willenborg and Scholtus (2018).

Section 4 we consider one of the software tools that we built to perform the bootstrapping for the SPAR index, namely the bootstrap sampler for SPAR (BSS). This is one of the two bootstrapping tools that we developed for the SPAR index. The other tool is the bootstrap collector for SPAR (BCS), which is discussed in Section 5. This tool combines the information in each of the bootstrap files generated by the BSS, and computes variances and biases. Section 6 discusses the results of some of the bootstrap computations that we performed. Chapter 7 briefly discusses some efficiency issues, in particular how to speed up the bootstrap computations for the SPAR index. These are just recommendations based on the type of problem at issue. They are not based on actual experiences. We had no time to consider parallelization. In Section 8 we close the main part of the paper by discussing results and insights obtained when working with the data. A brief list of references as well as an appendix complete the paper. Appendix A contains the R code that was used, in particular for the BSS and the BCS.

2 The input data

2.1 Form and contents

The input data is in the form of a matrix or rectangular file, where the rows (or records) correspond to houses and the columns to variables. There are 22 variables in this file. Some of these are of particular importance for our discussion: the variables that are used to stratify the population of houses (type of object and municipality) and those that indicate the period (year-month) when the houses were sold. Of key importance are the variables concerning selling prices and WOZ valuations of houses.

2.2 Missing data and outliers

Currently the computation of the SPAR index is somewhat complicated because of the occurrence of missing data (for WOZ valuations)[4] and outliers (for selling prices or WOZ valuations).[5] The input file contains variables that indicate which value in a record is considered an outlier. These variables are of importance in the bootstrapping procedure because it has to be made sure that a usable sample is drawn from the data: there should not be too many records with missing values. In Willenborg and Scholtus (2018) outlier detection is discussed.

In Willenborg and Scholtus (2018) it is also indicated that overlapping months and non-overlapping months need to be distinguished as they are treated somewhat differently in the SPAR index. For a detailed description of the (non-)overlapping months, the reader is referred to that paper. For the present discussion it is sufficient to note that selling prices have to be compared with two WOZ valuations. In non-overlapping months there is only one WOZ valuation available. This may cause a selling price in an overlapping month to be an outlier with respect to one of the two WOZ valuations but not with respect to the other one.

[4] Missing values for selling prices also occur, but they do not make it to the input file. They are filtered out at an earlier stage of the data processing of the house price data.

[5] Outliers are determined by applying certain criteria (edit rules) to the selling prices and WOZ valuations. For the bootstrapping procedure it is important to know whether these outlier criteria can be applied to each record independently and do not contain parameters that are computed from the observed data. If they do, outlier detection should be part of the bootstrapping procedure and the resulting bootstrap files may depend on them.

2.3 A closer look at the input data

The input data that we have used contains information on selling prices and WOZ valuations of two types of houses ('object types') in two regions (municipalities) over a period of 22 years (1995 - 2017). The input data set contains about 225,000 records in total, where each record represents a sold house.

We have analyzed some aspects of this input data set, in particular concerning the appearance of regular values and missing values for selling prices and WOZ valuations. Our interest was focused in particular on missings and outliers in the strata for each year-month combination. The missings that occur in the original input data are the result of processes beyond the control of CBS. They are referred to in the present paper as 'original missings'. Furthermore there are also missings as a result of the elimination of outliers. These missing are within the control of CBS and in fact dependent on the edits applied. See Willenborg and Scholtus (2018) for a discussion on this topic. The parameters and parameter values in these edits define which values are considered outliers.

As it turns out, the majority of the missings are original missings. These have been plotted in a series of Figures, viz Figures 2.1, 2.2, 2.3, 2.4 and 2.5. Figure 2.1 is intended to facilitate the comparison of results between the two regions and the two object types. The Figures 2.2, 2.3, 2.4 and 2.5 are the same as those used in Figure 2.1, but they have been rendered at a larger scale. In all graphs shown the total number of records available is indicated by a black line, the number of original missings by a red line, and the number of missings due to outliers are indicated by a blue line. In the discussion below it is sometimes useful to distinguish between types of missings and sometimes it is not. A record in the input file with original missings is called incomplete; otherwise it is called complete.

In these figures information is given about the number of complete and incomplete records per period, for two regions ('344' and '518', corresponding to the municipalities of Utrecht and The Hague, respectively) and two object types ('A' = apartments and 'T' = terraced houses, with the exclusion of end-of-terrace houses).

We make some observations about these data. First, by looking at Figure 2.1 and comparing the four situations shown there, and then by zooming in on each of them separately. In all four cases the number of complete records is higher than the number of incomplete ones. There are more objects of type A in region 518 than in region 344. For most of the time the number of incomplete records is fairly stable in each of the four cases, although at different levels. For region 344 and object type A, they are quite high, around 50. In the same region the number of incomplete records for object type T is much smaller and pretty constant over the 20 years time period. The number of incomplete records for region 518 is lower than for region 344 and object type A, but higher than for object type T in both regions. In Region 344 and object type A there is a rise in this number around 2015, and then a sharp drop. In region 518 there is a rise in the number of incomplete records between 2005 and 2008, for both type of objects, A and T. But apart from these anomalies, what is striking is that the number of incomplete records for each region and object type fluctuates somewhat, but not much. This observation is of importance to the bootstrapping: we do not want these fluctuations to be (much) bigger in the bootstrap files than they are in the input file. To achieve this, they should be explicitly controlled.

Now we take a closer look at each of the four figures represented in Figure 2.1. The first one is Figure 2.2 about object type A in region 344. Roughly between 2010 and 2015 we see a dramatic drop in the number of records, followed by a rapid rise at the end of this period. The same pattern is visible in the number of incomplete records, although at a lower level. At one period the sample size appears to be almost the same as the number of incomplete records, but in fact the peak in the graph of the number of missings and the valley in the graph of the sample size occur at different months. Also remarkable is the sharp drop in the number of incomplete records after 2016. All four figures reflect the financial crisis that started in 2008, with a drop in the number of houses sold, and a rapid recovery in 2015 when this crisis ended. A similar pattern is noticeable for both object types of region 344 and object type T in Region 518: in these three cases the number of house sales in 2015 is larger than before the crisis. In case of region 518 and object type A, however, we do not see such a strong recovery in house sales.

The next Figure is Figure 2.3 about object type T in region 344. Remarkable is the relatively small number of incomplete records over the entire period. This number is much smaller than the number of complete records, although it changes considerably over time.

Looking at Figure 2.4 about region 518 and object type A. Here the number of incomplete records is well below that of the complete records. This number is fairly stable, except for two periods, as was already remarked. As the same pattern is noticeable in Figure 2.5, so it seems reasonable to attribute it to the region. It is probably related to the administrative procedure handling selling prices and WOZ valuations.

The fourth figure is Figure 2.5 about region 518 and object type T. In this case the 'plateau' of incomplete records between 2006 and 2008 is remarkable. Also striking is the lower number of records in the period just before this plateau (from 2002 - 2006) and after (from 2008 - 2016). In about 2008 the number of incomplete records was a little bit lower than the number of complete records.

The main lesson from these results is that, generally speaking, the number of incomplete records is fairly constant, at a level that depends on region and object type. However, there are periods where the number of incomplete records may be noticeably higher, and even coming close to the number of complete records. So there is reason to control the bootstrap samples to make sure that the number of incomplete records in a bootstrap sample does not get out of hand. A simple way to achieve this is to fix the number of complete records per period - stratum combination that is present in the input file.

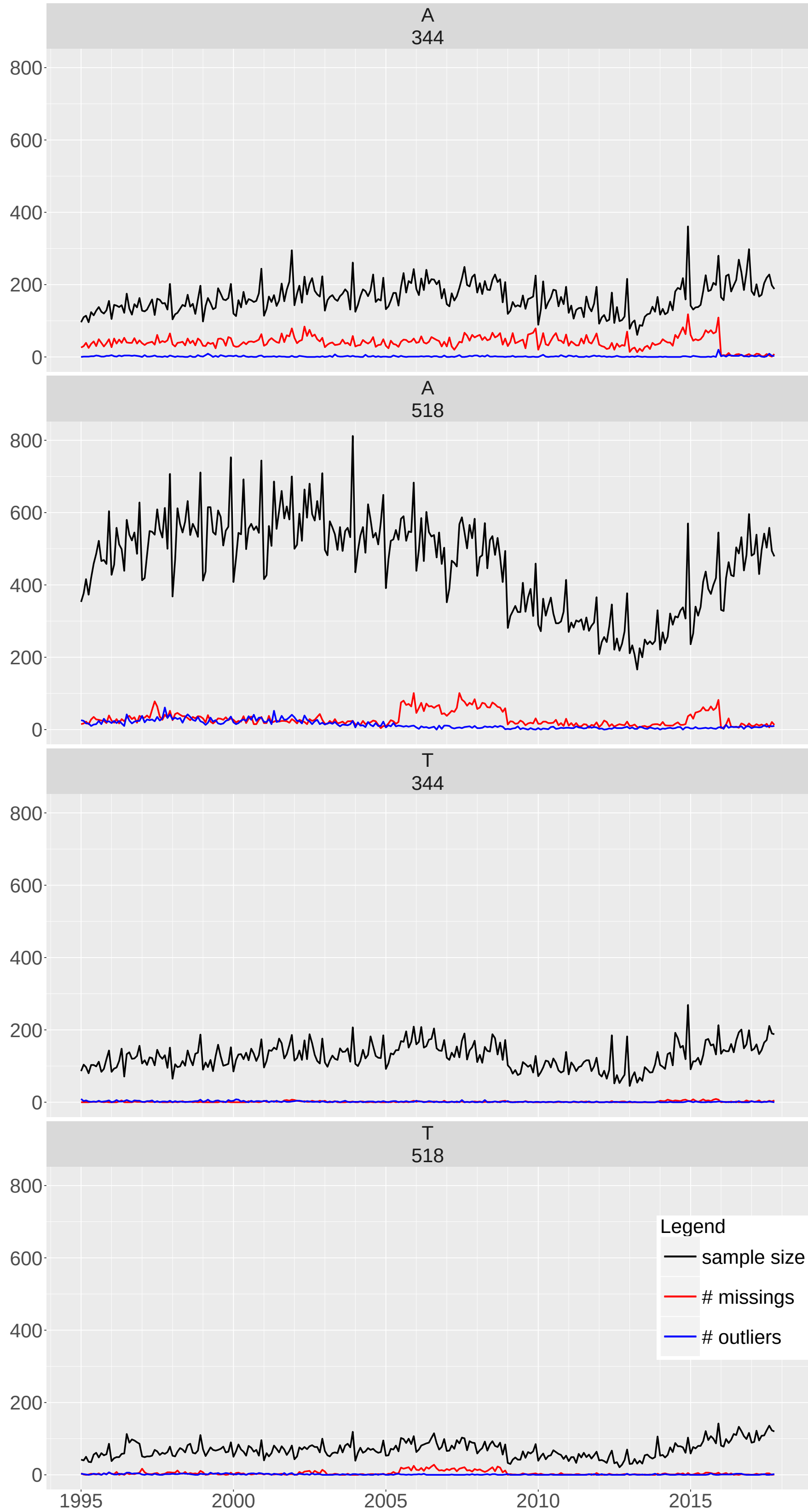

Figure 2.1 Number of complete and incomplete records (with original missings and missings due to outliers) in two regions (344 and 518) and for two object types (A = Apartments and T= Terraced houses).

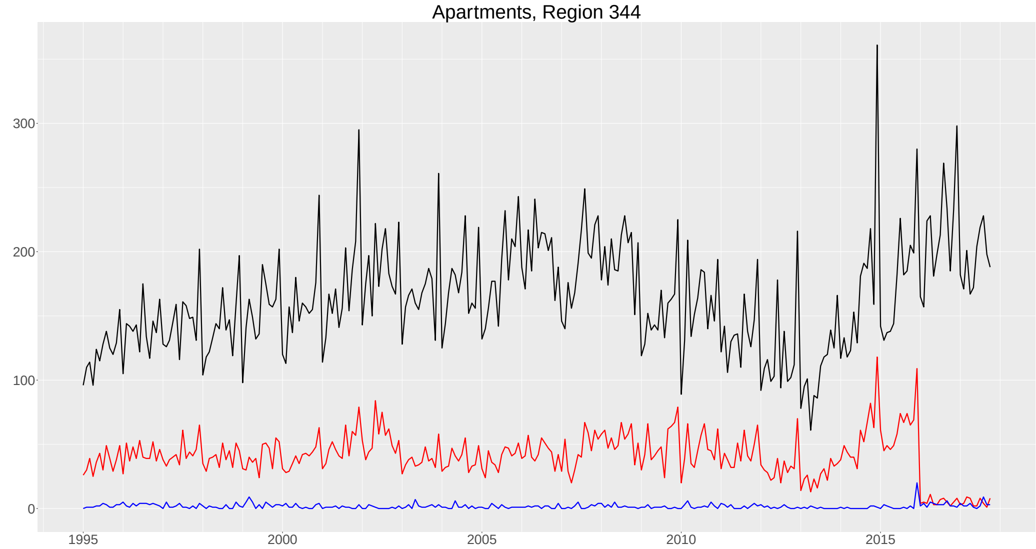

Figure 2.2 Number of complete and incomplete records (with original missings and missings due to outliers) in region 344 and object type A.

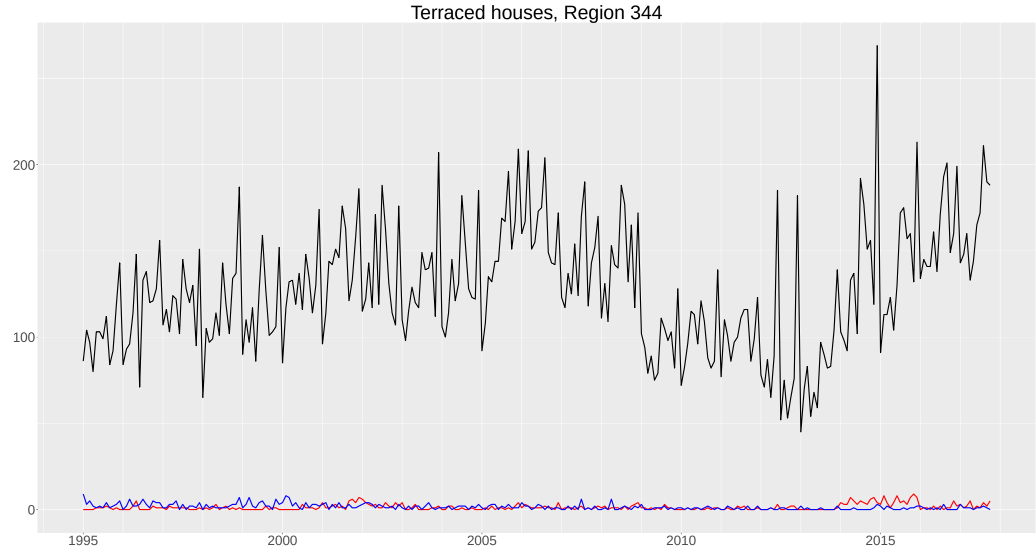

Figure 2.3 Number of complete and incomplete records (with original missings and missings due to outliers) in region 344 and object type T.

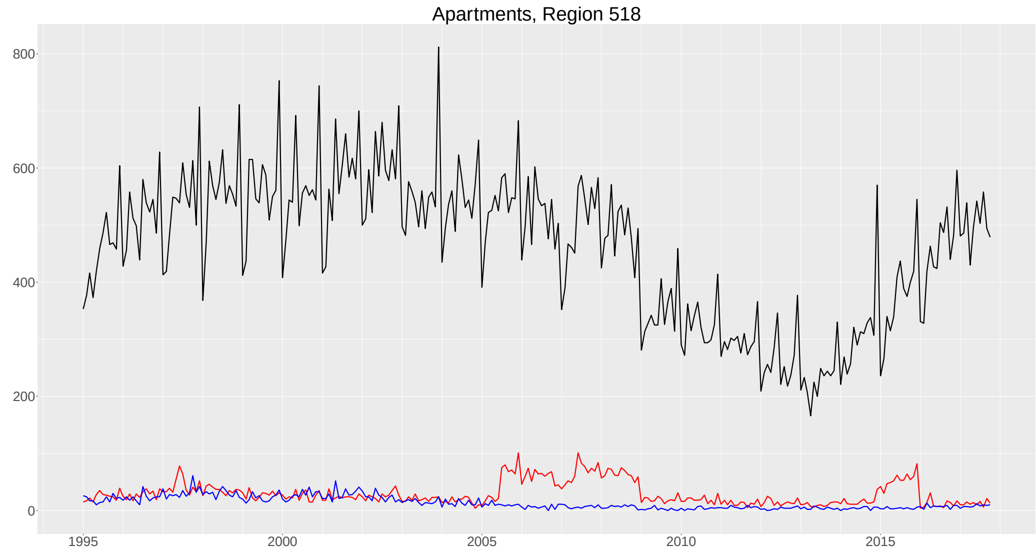

Figure 2.4 Number of complete and incomplete records in region 518 and object type A.

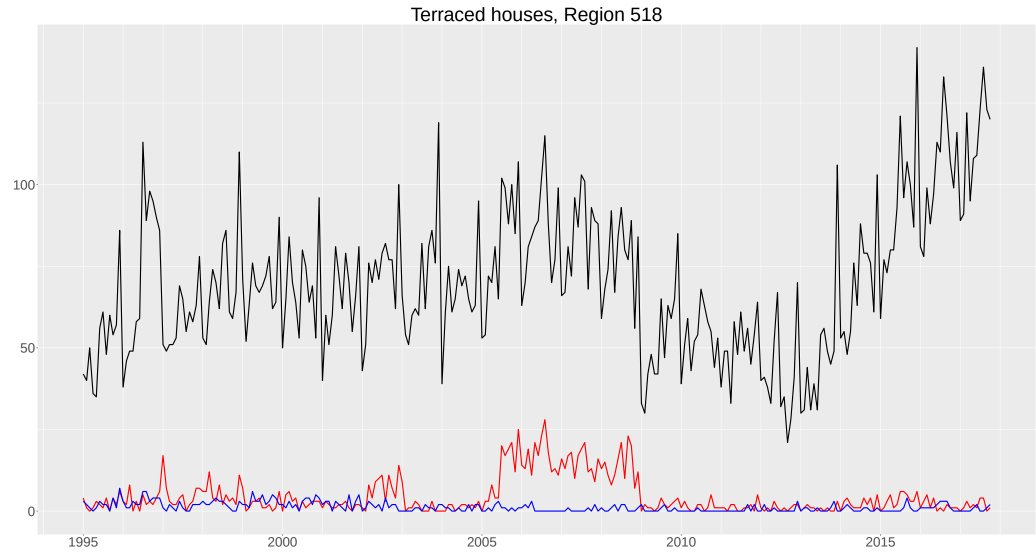

Figure 2.5 Number of complete and incomplete records in region 518 and object type T.

3 Bootstrapping the SPAR index

3.1 The bootstrapping method in general

Bootstrapping is a computer intensive technique that is used - crudely speaking - to get an idea of the variability of an estimator by using variations of the input data. These variations are obtained by sampling with replacement from the input file. The method is extensively described in the statistical literature, for example in Davison and Hinkley (2009) and by Efron and Tibshirani (1994).

The bootstrapping method was originally developed for data drawn from infinite populations, with continuous distributions. Applications to finite-population data have been implemented later. In a sense, these are more complicated than those used in the original situation. This is a result of the more realistic character of finite sampling situations: sample designs can be more complicated, there can be non-response and errors (like outliers) in the data, etc. All these factors complicate the application of the bootstrap method to applications with samples from finite populations. These complications are missing in the case of continuous probability distributions, for which the original bootstrap procedure was designed.

3.2 Extreme bootstrap samples

The strata that are to be bootstrapped contain original missing values and induced missing values (by replacing outliers by missings). Records with a missing value are not used in the current SPAR computations.

Suppose we have a stratum in a particular period with incomplete records in the input data. Because the bootstrapping procedure takes independent samples of the records in the stratum, it is, in principle, possible that a sample is drawn that exclusively consists of incomplete records. In such a case a SPAR index value cannot be computed. This is a situation that we would like to avoid. We want to get an idea how large this probability is. In case of a stratum g of size n with $m \leq n$ incomplete records the probability that such a sample is produced equals $\left(\frac{m}{n}\right)^n$. We would like this probability to be smaller than some upper-bound $0 < \gamma \leq 1$:

$$\left(\frac{m}{n}\right)^n \leq \gamma. \qquad (1)$$

In Table 3.1 for various values of n and γ the critical values of m are shown.

As Table 3.1 shows only for small values of n one can run into trouble. For sample sizes $n = 100$ the percentage of incomplete records can be 85% or more to yield 'nasty' samples, with only incomplete records, with a probability that is negligible for all practical purposes.

$n \setminus \gamma$	0.01	0.001	0.0001	0.00001	0.000001	0.0000001
1	0.0	0.0	0.0	0.0	0.0	0.0
2	0.2	0.1	0.0	0.0	0.0	0.0
3	0.6	0.3	0.1	0.1	0.0	0.0
4	1.3	0.7	0.4	0.2	0.1	0.1
5	2.0	1.3	0.8	0.5	0.3	0.2
6	2.8	1.9	1.3	0.9	0.6	0.4
7	3.6	2.6	1.9	1.4	1.0	0.7
8	4.5	3.4	2.5	1.9	1.4	1.1
9	5.4	4.2	3.2	2.5	1.9	1.5
10	6.3	5.0	4.0	3.2	2.5	2.0
15	11.0	9.5	8.1	7.0	6.0	5.1
20	15.9	14.2	12.6	11.2	10.0	8.9
30	25.7	23.8	22.1	20.4	18.9	17.5
40	35.7	33.7	31.8	30.0	28.3	26.7
50	45.6	43.5	41.6	39.7	37.9	36.2
60	55.6	53.5	51.5	49.5	47.7	45.9
80	75.5	73.4	71.3	69.3	67.3	65.4
100	95.5	93.3	91.2	89.1	87.1	85.1

Table 3.1 Critical values for m for given values of n and γ.

3.3 SPAR tool

The SPAR tool is used to compute the SPAR results. It was written in R by the department responsible for compiling price statistics for houses and commercial property. This tool is used 'as is' for our bootstrap computations.[6)] In fact our concern is that bootstrap samples are generated that on the one hand show sufficient variation and on the other can be processed without problems by the SPAR tool. The missing values in the data are in fact the nuisances that need to be controlled in the bootstrap sampling so that the SPAR tool can work properly.

4 BSS: Bootstrap sampler for SPAR

4.1 Purpose of BSS

The bootstrap sampler for SPAR is - as the name suggests - a tool to produce bootstrap samples which can be used as input files for the SPAR tool. So it is especially designed for this particular application. The main problem that it has to cope with is the presence of missings. It should be able to produce bootstrap samples that have enough variation on the one hand and on the other it should produces bootstrap samples that do not create problems for the tool to compute the SPAR index.

One way to do this is by working with bootstrap files that are also incomplete, in case the input file is incomplete. In this case we should control the number of incomplete records.

6) Except that only those parts were used that we needed for our bootstrapping. But this was a simple copy-paste procedure. The code we used, however, was as delivered to us. We did not modify the parts we used. This approach should facilitate the bootstrapping for the SPAR index.

Another option is to work with bootstrap files that have been completed by imputing the missing values first. As the imputation procedure applied is based on the input data (and no external data), the imputation procedure is part of the bootstrap procedure. The approach is: draw a bootstrap sample and then impute the missing values it contains using data from the bootstrap file. The completed bootstrap file is in the same format as the input file and can be used directly by the SPAR tool.

Both approaches - with and without imputation - are discussed in the present section.

4.2 Handling outliers

Outliers in non-overlapping months and in overlapping months are treated differently in the bootstrapping procedure. In the non-overlapping months we are dealing with only one WOZ valuation and whether or not a selling price is an outlier depends only on the selling price and corresponding WOZ valuation. We can simply use the indicator function for outliers and do not have to check anything.

In case of an overlapping month, we have, for each record, a selling price and two WOZ valuations. It has to be decided with respect to each of these WOZ valuations whether or not the selling price and the WOZ valuation combination satisfy or violate the outlier criterion.

The bootstrap procedure is slightly different for non-overlapping and overlapping months, depending on the number of WOZ valuations to consider: one or two. In both cases, we make sure that we bootstrap the same number of complete records in each stratum in the input file.

In case of a non-overlapping month we simply look at the values for the outlier indicator for this WOZ valuation to find out which of the complete records do or do not contain outliers. We use these indicators for the SPAR tool. In case of an overlapping month we use the complete records to find out to which of the 4 possible categories they belong: no outliers, 1 outlier (2 possibilities), 2 outliers. In the first three cases at least one WOZ valuation can be used. Only in the last case the record should be discarded, as it does not contain usable price information.

We assume that, in case of a violation of an outlier criterion, the WOZ valuation is the outlier and not the selling price.[7)] For each of these four categories it is clear for which computations in the SPAR tool they are suitable.

4.3 Incomplete input data

In agreement with the approach taken in the current practice concerning the SPAR index, the first idea is to apply the bootstrap using incomplete data. This makes sense. But a drawback of this approach is that it complicates the bootstrap computations.[8)] In particular one runs the risk, if not managed well, that the bootstrap samples generated may be unusable for the SPAR tool

7) This is to simplify our computations. In a more sophisticated approach closer examination of WOZ valuation and selling price should lead to a decision which of the values should be considered to be 'outlying'. For our experiments we have side-stepped this question.

8) If the idea behind this approach is to be neutral with respect to the missing values, we can remark that it is not. Not using missing data is also a choice. And applying imputations is another one, and not necessarily a worse one.

because they contain too many missing values. This problem was discussed in Section 3. Apart from the solution suggested there one could also use rejective sampling. In this approach samples are generated without any restrictions. Each sample is then checked if it is usable. If it is, it will be used. If not, it is rejected and a new sample is drawn. This method works well if the probability of rejecting a sample is not too big.

4.4 Completed input data

Instead of working with incomplete data (i.c. missing WOZ valuations)[9)] one can also try to complete the data by using imputation and then bootstrap the completed data set. In Willenborg and Scholtus (2018) several methods are suggested to impute missing WOZ valuations. The imputation method used is based on the ratio of selling prices and WOZ valuations. Of course, the imputation should also be part of the bootstrap procedure, as it is based on data from the original, incomplete data. We refer the interested reader to Willenborg and Scholtus (2018) for more information on the imputation methods proposed and the consequences this extra step has for the bootstrapping.

4.5 Internal workings of BSS

We give a general description of how the BSS works. It should give the reader a good idea as to how it works, leaving the details aside. If one is interested in this, the R code of this tool should be consulted; see Appendix A.

The BSS is supposed to produce bootstrap sample files from the input data set in the same format and with the same number of records as the input file. If there were no records with missing values, simple random samples with replacement are drawn from the input file, per stratum and for each period. But in case there are records with missing values the idea is to replicate the *number* of records with missing values. This is achieved by partitioning each stratum into a few substrata, for each period. These are the following ones: the one consisting of records without missings, the one consisting of records with selling prices available and WOZ valuation missing.[10)] BSS actually samples a period - stratum combination separately and makes sure that, in each period, the substrata have the same size as in the original data set. This limits the variability in the samples obtained. This is justified if we look at the input data (cf. section 2.3). BSS does not sample the records of the input file directly, only the record numbers, for which it keeps the multiplicities. When the sampling is finished the bootstrap samples are produced. This is done by first sorting a copy of the input file, per period - stratum combination, with respect to the multiplicities. Then the portions of records are taken with the same number of multiplicities m and such a portion is copied m time to the bootstrap sample file that is being constructed. Such a file is then written as a csv file.[11)]

As sampling is used in this tool, some care should be given to the use of random generators. This concerns the choice of a seed. Sometimes one would like to recompute a problem and one

9) Or selling prices. But they are not in the input data we have analyzed.
10) Records with selling price missing do not occur in the input data that we have used. This would involve two substrata, depending on whether the WOZ valuation is missing or not.
11) In theory. But this would be rather slow in case the input file is as sizable as in case of the house prices data. So this is done internally. See subsection 7.2.

should be able to compute the exact same bootstrap sample file as was used before. At other occasions rather the opposite should be the case and one wishes to compute a new bootstrap sample, independently of those that have been drawn before.

5 BCS: Bootstrap collector for SPAR

This tool is mainly administrative and collects the statistical information that is of interest in our bootstrap computations from each output file of the SPAR tool . It also computes bootstrap estimates from the collected data. These computations are quite easy.

The primary quantities BCS computes are variances and biases. With these a number of other interesting quantities are actually computed (confidence intervals) or could easily be added to BCS (mean square arror, MSE). It is assumed that that simple random sampling with replacement is used, with fixed sample of size n. Let Y be an unknown population quantity (a total) for which $\hat{y}$ is an estimator. The expressions for the variance and the bias that BCS uses are:

$$\text{Var}(\hat{y}) = E(\hat{y} - E\hat{y})^2, \tag{2}$$

$$\text{Bias}(\hat{y}) = E\hat{y} - Y. \tag{3}$$

From (2) and (3) it is easy to compute the mean square error:

$$\text{MSE}(\hat{y}) = E(\hat{y} - Y)^2 = \text{Var}(\hat{y}) + \text{Bias}^2(\hat{y}). \tag{4}$$

MSE is actually not computed in BCS, but it is easy to add.

Suppose the BSS has been applied B times to obtain estimates $\hat{y}_1, \ldots, \hat{y}_B$ in addition to the estimate $\hat{y}$ based on the original input data. A variance estimator for (2) based on these bootstrap outcomes is:

$$\widehat{\text{Var}}(\hat{y}) = \frac{1}{B-1} \sum_{i=1}^{B} \left(\hat{y}_i - \frac{1}{B} \sum_{j=1}^{B} \hat{y}_j \right)^2 . \tag{5}$$

A bootstrap estimator for the bias (3) is:

$$\widehat{\text{Bias}}(\hat{y}) = \frac{1}{B} \sum_{i=1}^{B} (\hat{y}_i - \hat{y}) = \frac{1}{B} \sum_{i=1}^{B} \hat{y}_i - \hat{y}. \tag{6}$$

An estimator for the mean square error is obtained by substituting (5) for (2) and (6) for (3) in expression (4).

Finally, a confidence interval for Y can be obtained from the bootstrap method in several ways. We will discuss two basic techniques to do so. Firstly, if we can assume that the distribution of the estimator $\hat{y}$ under repeated sampling is approximately normal - which will be true by the central limit theorem if n is sufficiently large - and Bias($\hat{y}$) is negligible, then a $(1-\alpha) \times 100\%$ confidence interval for Y can be obtained using (5):

$$\hat{y} - q_{1-\alpha/2}\sqrt{\widehat{\text{Var}}(\hat{y})} \leq Y \leq \hat{y} + q_{1-\alpha/2}\sqrt{\widehat{\text{Var}}(\hat{y})}, \tag{7}$$

where $q_{1-\alpha/2}$ denotes the $(1-\alpha/2) \times 100\%$ quantile of the standard normal distribution. For instance, to obtain a 95% confidence interval, we choose $q_{0.975} \approx 1.96$. Secondly, a confidence interval can be obtained from the empirical bootstrap distribution. Arrange the B bootstrap outcomes from smallest to largest: $\hat{y}_{(1)} \leq \hat{y}_{(2)} \leq \cdots \leq \hat{y}_{(B)}$. Then a $(1-\alpha) \times 100\%$ empirical bootstrap confidence interval for Y is given by:

$$\hat{y}_{(L)} \leq Y \leq \hat{y}_{(U)}, \tag{8}$$

where L is the nearest integer to $\alpha/2 \times B$ and U is the nearest integer to $(1-\alpha/2) \times B$. Unlike (7), the interval (8) need not be symmetric around $\hat{y}$. For large values of n and B, the two intervals should be in close agreement.

BCS computes the quantities (5), (6), (7) and (8), as one can verify in Appendix A.

6 Results

Here we have collected some of the results obtained from our bootstrap computations.

Figures 6.1 to 6.4 show the estimated long series for the SPAR index based on the original data, for each of the four strata, as well as a 95% confidence interval computed from $B = 500$ bootstrap samples. All confidence intervals shown in the figures below were obtained by the empirical method (8). We also computed intervals based on the normal approximation (7). As these were very similar to those computed with the empirical method, they are not shown here.

In Figure 6.5 results have been collected that should give an impression about the speed of convergence of the bootstrap procedure. In this figure the variance estimates in the various strata (region - object type) are shown for each month of one year (2015), as a function of the number of bootstrap samples. It is clear that after 300 samples the results do not change much. So this means that for the SPAR index computations there is no need to generate 1000 bootstrap

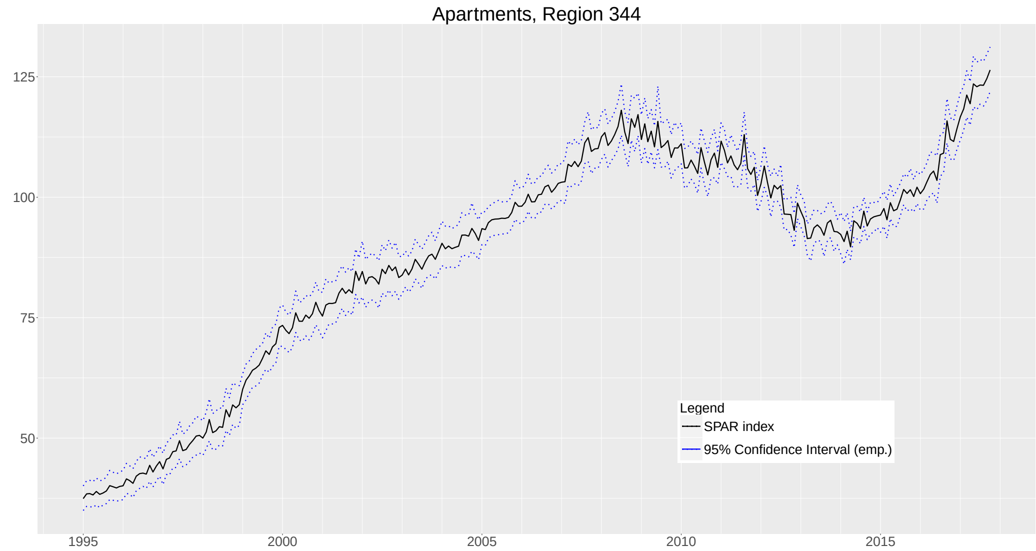

Figure 6.1 Long series for the SPAR index with estimated 95% **confidence intervals for region 344 and objecttype A. Base year is 2012.**

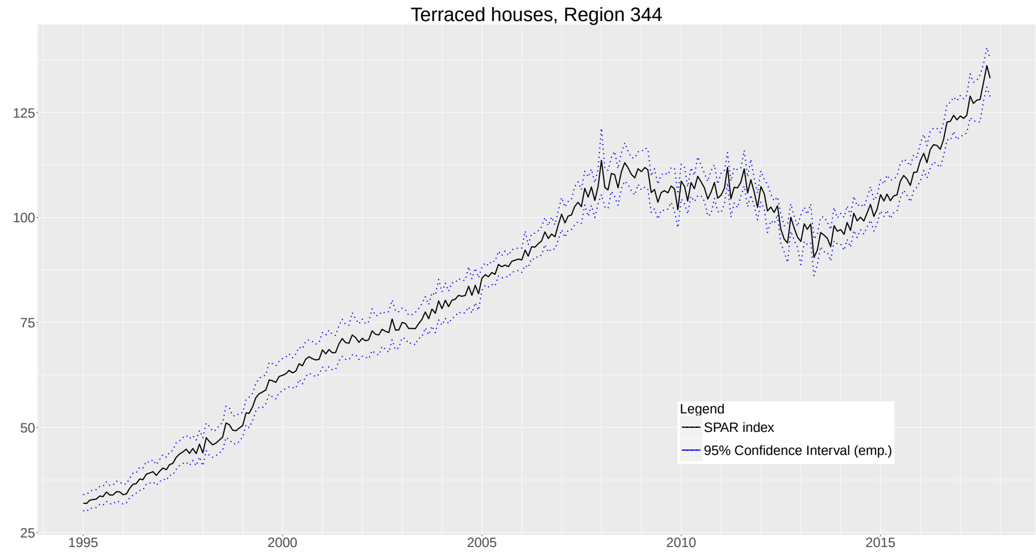

Figure 6.2 Long series for the SPAR index with estimated 95% **confidence intervals for region 344 and objecttype T. Base year is 2012.**

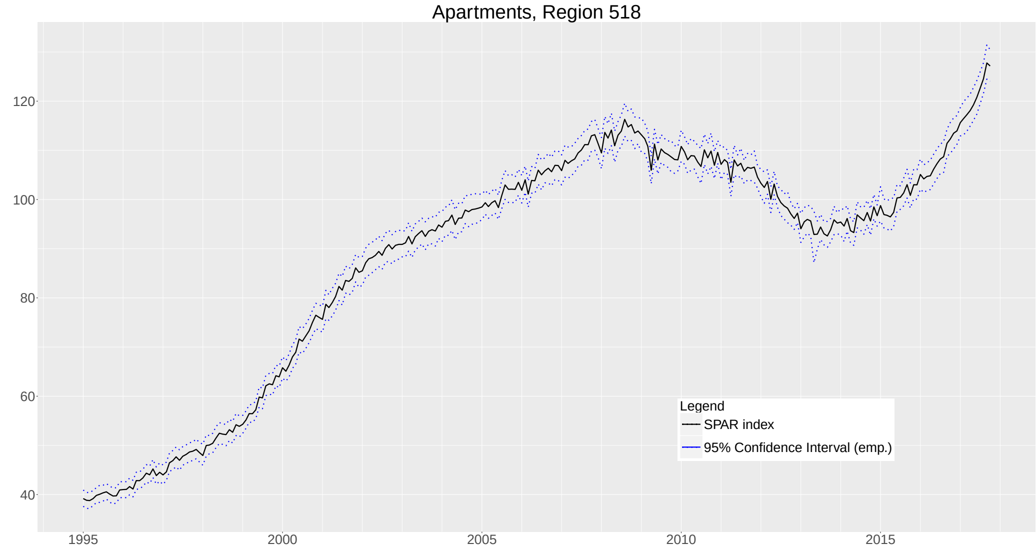

Figure 6.3 Long series for the SPAR index with estimated 95% confidence intervals for region 518 and objecttype A. Base year is 2012.

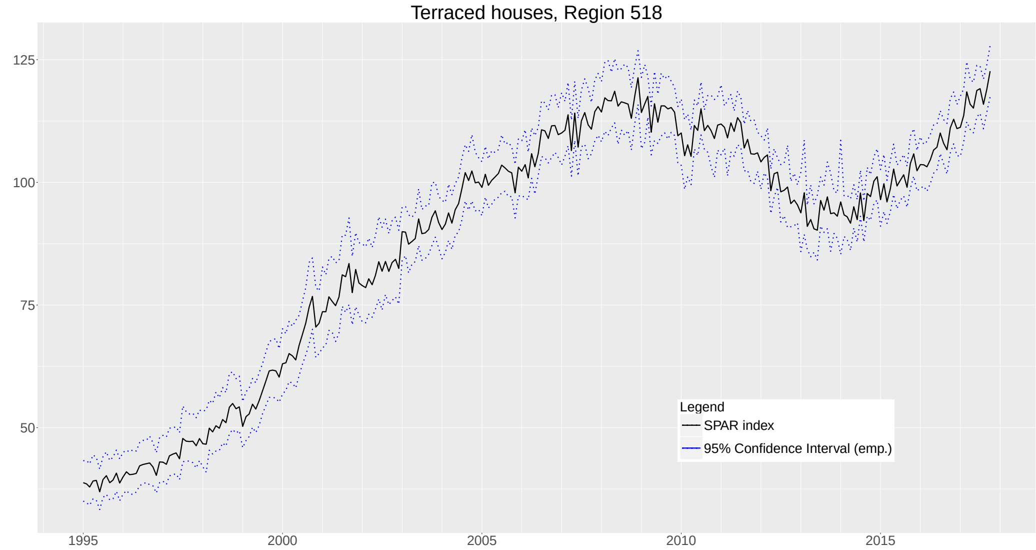

Figure 6.4 Long series for the SPAR index with estimated 95% confidence intervals for region 518 and objecttype T. Base year is 2012.

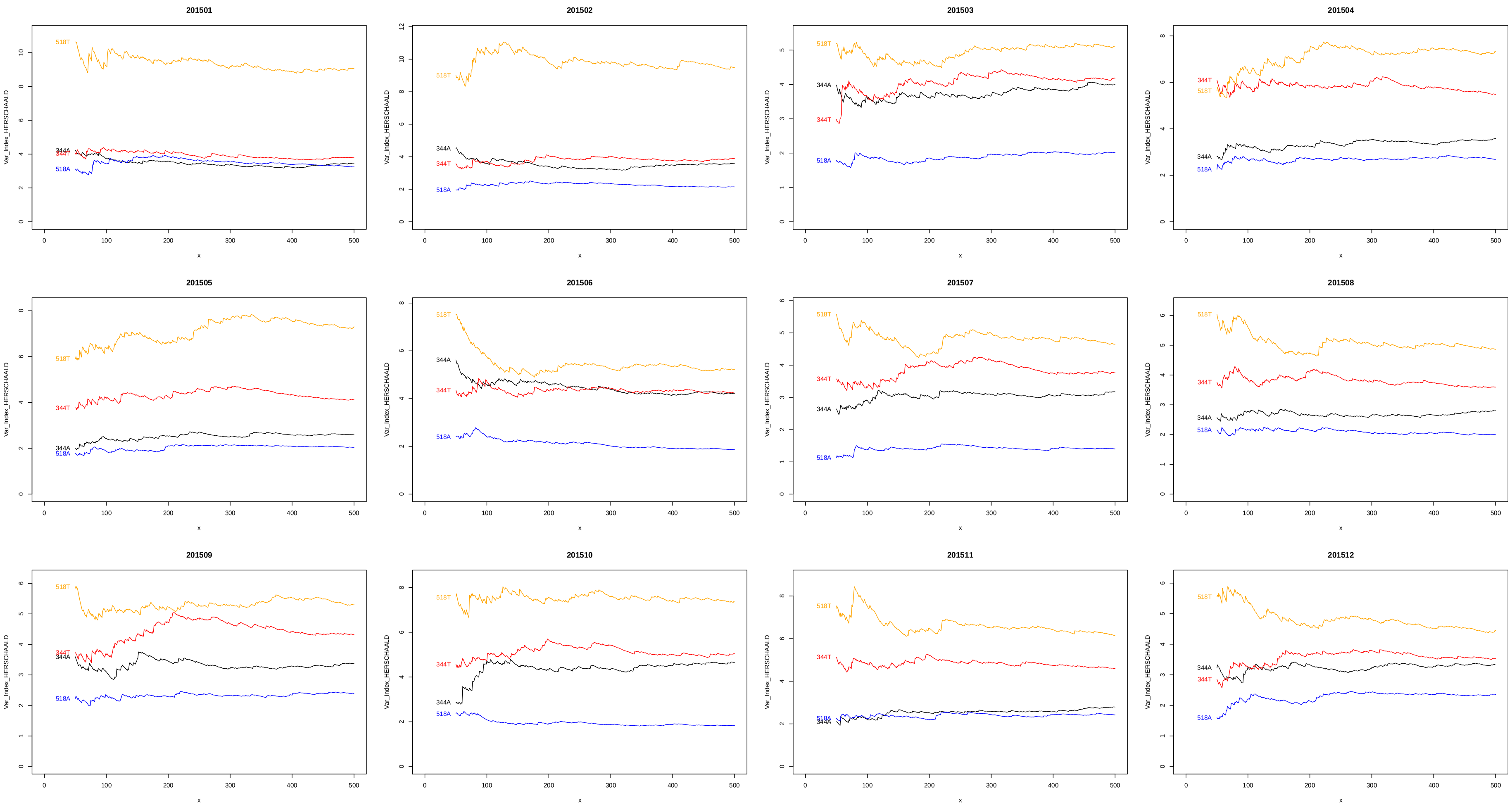

Figure 6.5 Convergence of the variance estimates as a function of the number of bootstrap samples, for each month in 2015 for regions 344 and 518 and objects T and A.

samples. About one third of this number would already be sufficient. From an efficiency point of view this is an interesting result.[12)]

In Figure 6.6 the matching factors for the overlapping months and the 95% confidence interval have been plotted for the two regions and two object types, for the entire period covered by the input data. These factors are used to match short index series to the long series for the SPAR index. So they are critical for the long index. See Willenborg and Scholtus (2018) for some background on these matching factors. Figure 6.6 shows that the confidence intervals differ in average length, depending on the region and type of object. Furthermore the factors can be seen to fluctuate around 1, without apparent bias in one direction. This is a good thing, as it prevents the long index from becoming upward or downward biased.

Figure 6.7 shows the coefficient of variation[13)] for the two regions and two object types, for the entire 20 years period. To understand it one should realize that the year 2012 was chosen as the base year for the long index. This means that the average index value for this year is fixed at 100%. The index values for the months in 2012 therefore have little room for variation, and the coefficient of variation for these months should be relatively small. As one gets further away from 2012 - in either direction - the coefficient of variation is expected to increase. This effect is clearly visible if one looks at the period prior to 2012. The same would have been visible for the period in the future, if the sale of houses had not increased dramatically in that period. And larger sample sizes lead to a reduced coefficient of variation. The total effect of these antagonistic influences is a development that stays more or less at the same level, except for some isolated peaks. Also notice that the picture reflects the fact that the WOZ valuations in the past were kept the same for 5 year periods; later the WOZ valuations were renewed on a yearly basis.

The next four figures, Figures 6.8, 6.9, 6.10 and 6.11, contain the short SPAR index series with 95% confidence interval. Contrary to the long index series, the short series start at 1 (100%) everytime a new WOZ valuation becomes available, that is, at the beginning of a new WOZ period. In the 1990s the length of a WOZ period was 5 years, in due course this was reduced to 1 year. The pictures clearly reflect these periods, especially at the beginning of the 20 year period. In each case 500 bootstrap samples were generated for the computations.

12) In practice, bootstrap variance estimates tend to converge faster when the original sample size is larger. In Figure 6.5 this can be seen to the extent that the blue lines - corresponding to object type A in region 518 which is the largest stratum - are the first to ''flatten out''. Theoretically, this can be explained by the fact that the additional variance in the bootstrap variance estimator due to using only a finite number of resamples B is of the order $1/B$, with a proportionality constant that increases with the expected kurtosis of the bootstrapped estimator $\hat{y}_i$; see Efron and Tibshirani (1994, Section 6.4). According to the Central Limit Theorem, this expected kurtosis will typically tend to zero as the original sample size n increases, leading to faster convergence of the bootstrap variance estimator.

13) Which is equal in this case to the square root of the bootstrap variance divided by the observed SPAR index value.

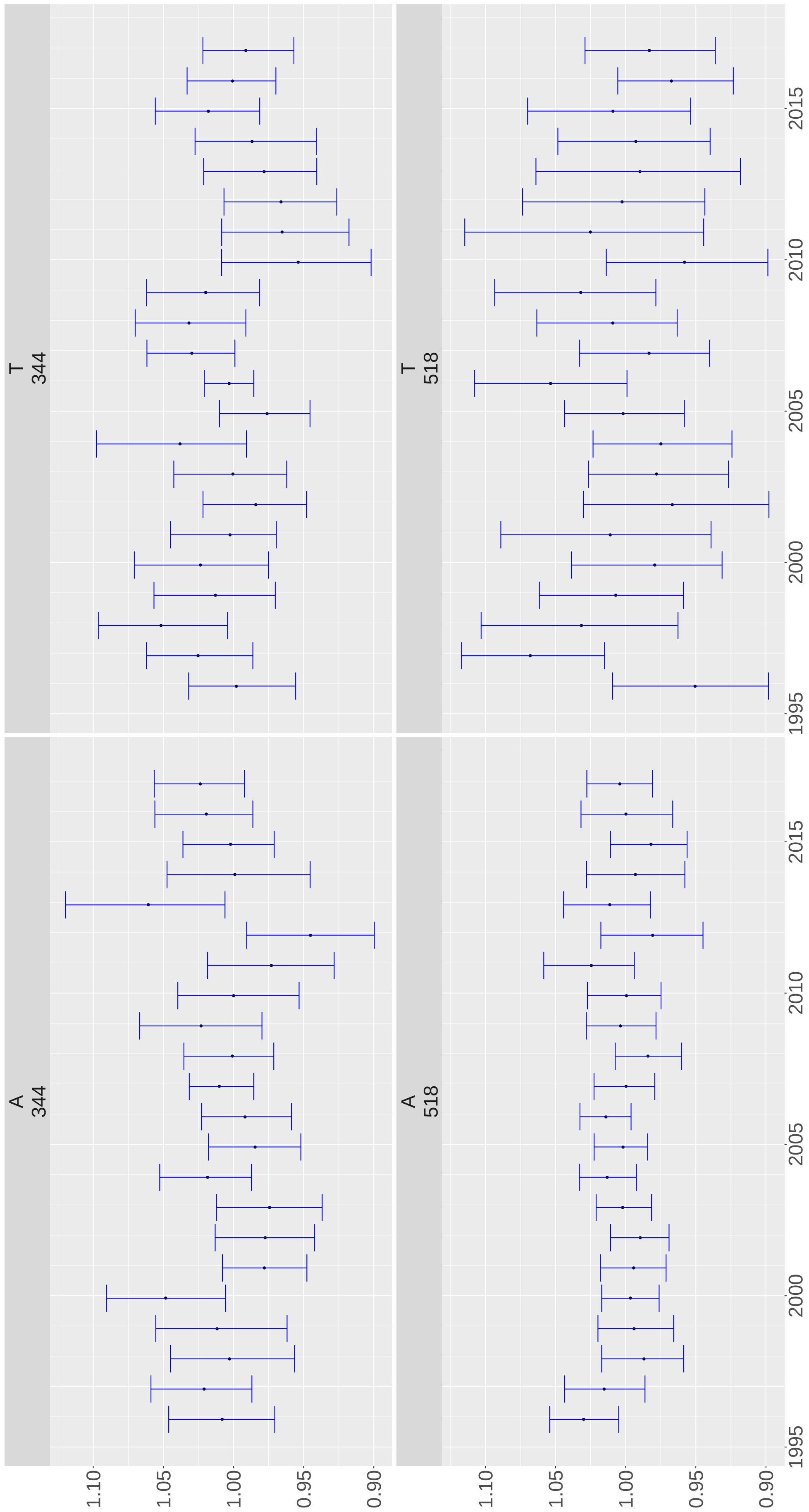

Figure 6.6 Matching factor and (empirical) 95% **confidence intervals for the overlapping months (December) for two regions and two object types.**

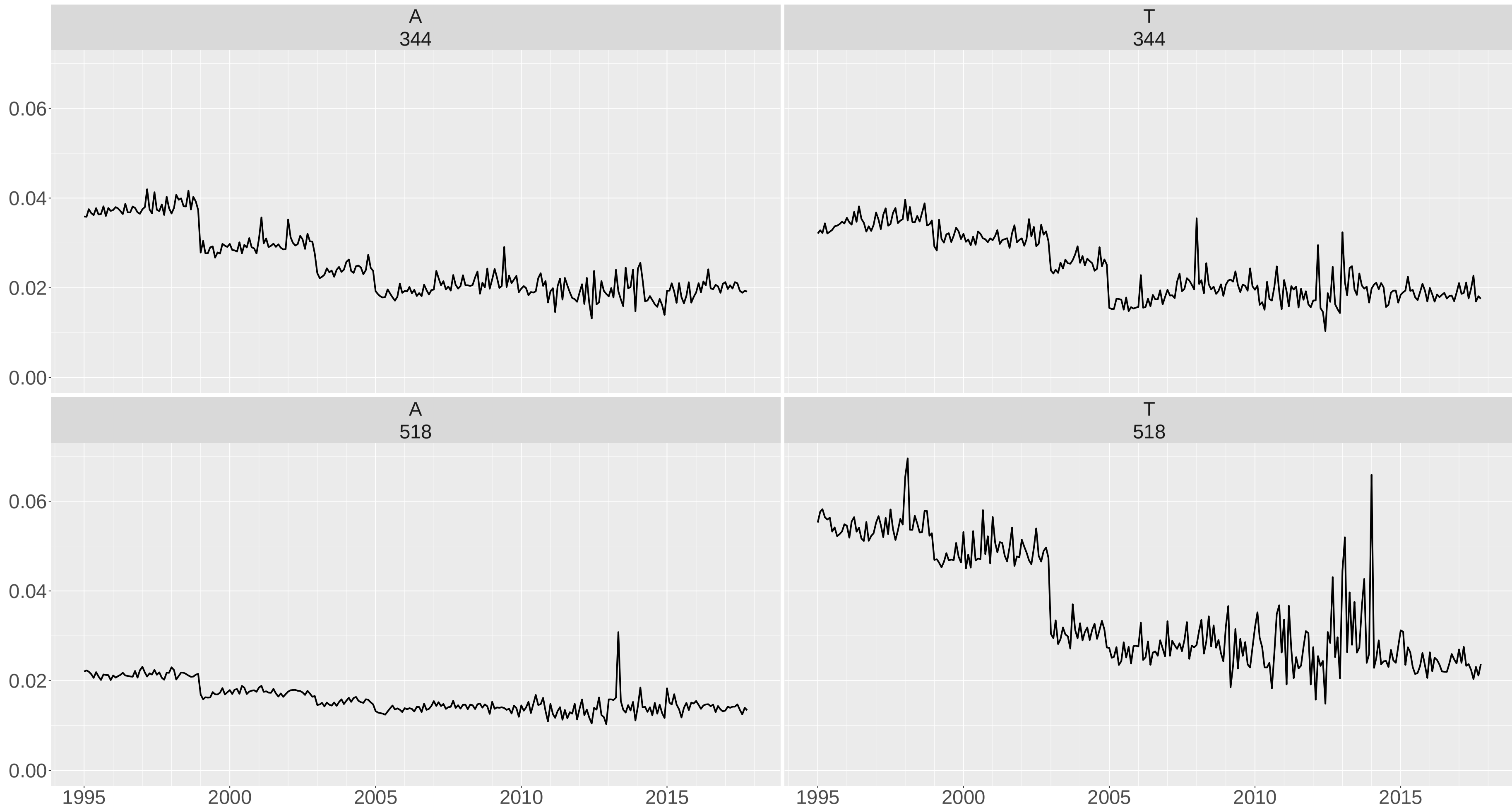

Figure 6.7 Coefficient of variation for the estimated long series for the SPAR index for two regions and two object types.

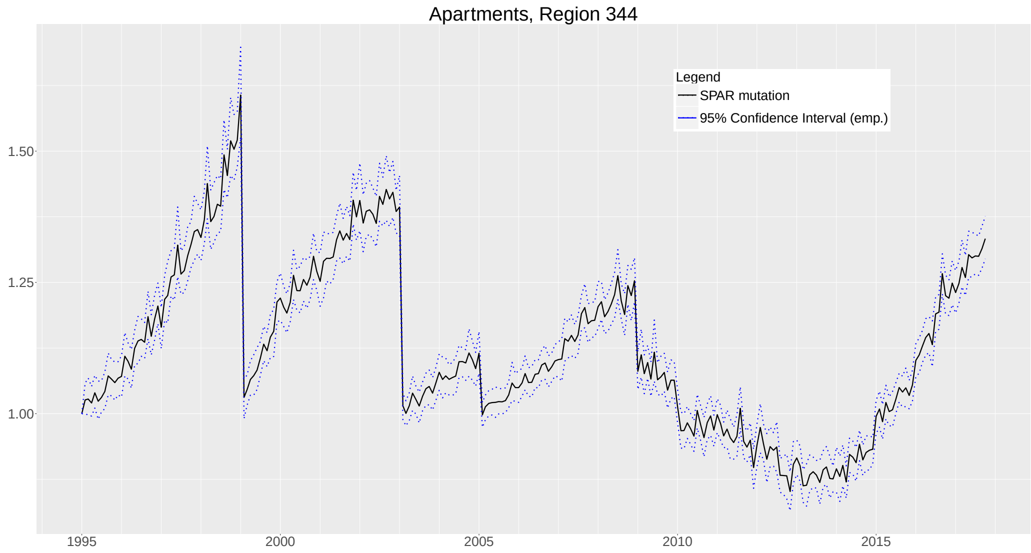

Figure 6.8 Short SPAR series for region 344 objecttype A.

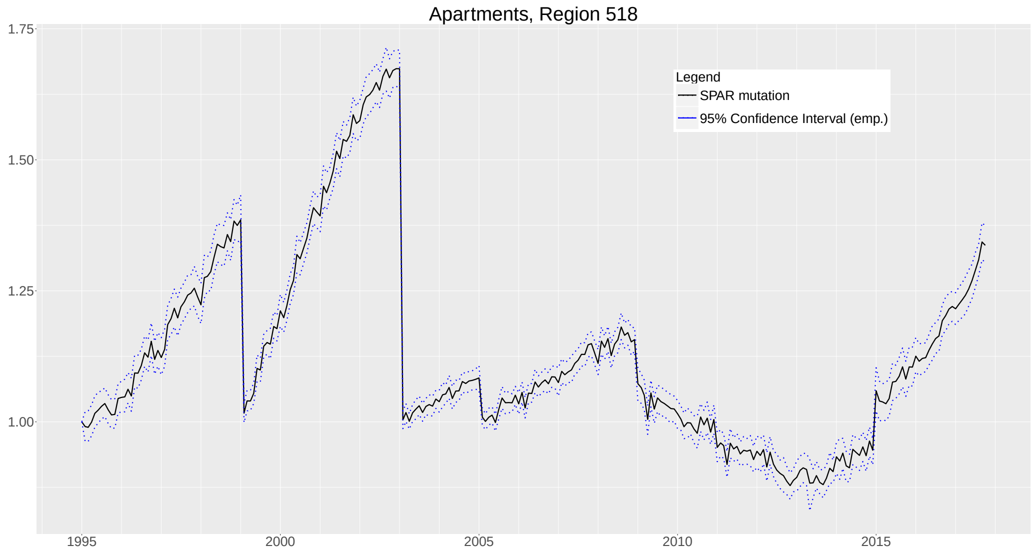

Figure 6.9 Short SPAR series for region 518 objecttype A.

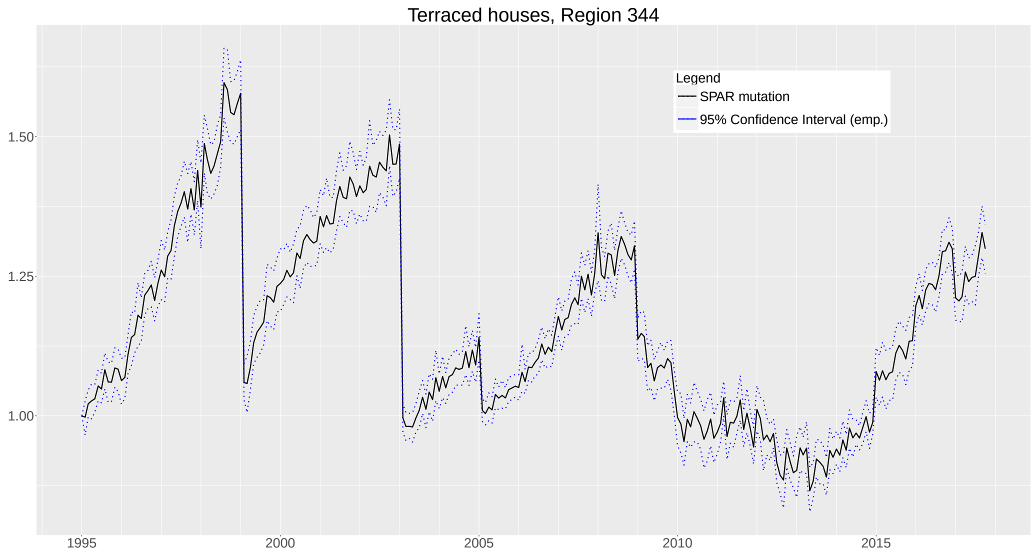

Figure 6.10 Short SPAR series for region 344 objecttype T.

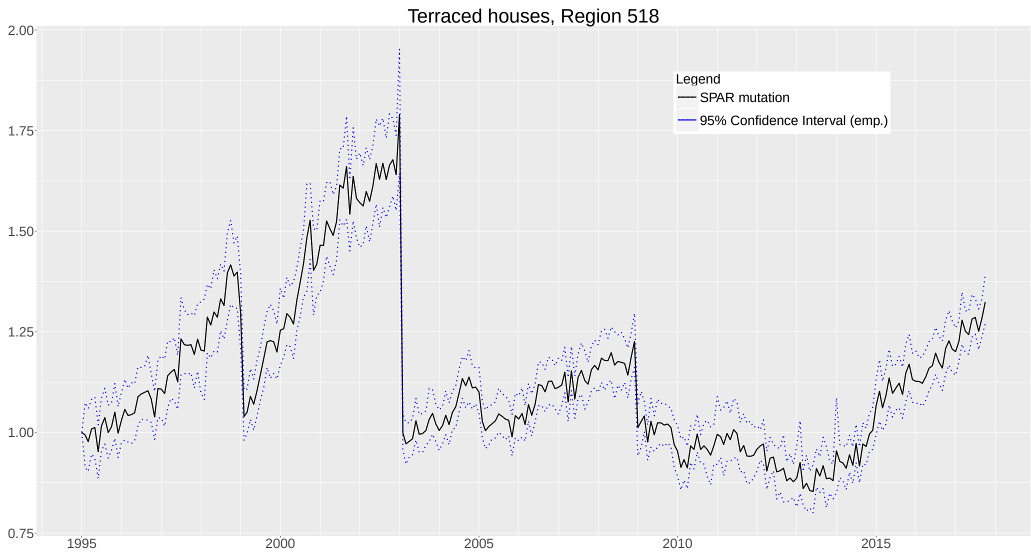

Figure 6.11 Short SPAR series for region 518 objecttype T.

7 Speed and efficiency

7.1 Number of bootstrap files

As was remarked in Section 6 the number of bootstrap files can be chosen to be in the range of 300. It is a more or less standard choice to take 1000 as a default value. But this number is not needed here as the results stabilize already at the smaller number.

7.2 BSS, BCS and the SPAR tool

In section 4 the BSS is presented as a separate tool that generates bootstrap sample files in the form of csv-files that are read and processed by the bootstrap tool. This is, however, a relatively time consuming process as it involves a lot of IO. In case the tool is a commercial piece of software, in compiled form, this is how one should proceed. But in our case, with the SPAR tool we are dealing with a piece of in-house made R code. This allows us to combine the R tool and BSS. The advantage is that the input file has to be read only once and the bootstrap files can be used 'internally' without the need of first writing them to a csv file, which in turn has to be read by the SPAR tool. This approach therefore saves expensive IO operations.

What has been said about the BSS and the SPAR tool also holds for the BCS and this tool: they can also be combined, so that the collection of bootstrap computations for each bootstrap file are kept internally and can be efficiently used to produce bootstrap estimates.

7.3 Parallel computation in R

Bootstrapping for SPAR and similar indices has the possibility to be carried out in parallel. The procedure is applied to separate samples, which can be processed independently and hence in parallel. The generation of the bootstrap samples is perhaps best carried out centrally, but the actual bootstrap computations can be carried out on different machines or processors.

As the computations are even independent per period-stratum combination, it would even be possible to carve up a bootstrap file into several sub files, each containing disjoint period-stratum combinations.

7.4 Cumulating bootstrap files

In practice the SPAR index is built in a cumulative way: new data constantly arrive and at the end of certain periods the index values for a new month can be computed. The same is true, in case updates have been received pertaining to previous months. In case the definitive figures of the SPAR index have been produced, one can also keep the bootstrap results for the corresponding period. There is no need to compute them again every time this 'definitive period' is extended. Bootstrap results only for the latest period need to be added to the collection of similar results for the previous period.

8 Discussion

One of the foci of the present paper is on the requirements for a useful BSS. This is of importance if bootstrap computations are to be performed on a regular basis, perhaps even as part of the standard output of the department of house prices. BSS can, in principle, be used by the department responsible for housing statistics and statistics of commercial property to be used in conjunction with their regular tool for computing the SPAR index. Or it can be used as an initial tool for this purpose.

The main difficulty for the BSS is the presence of missings. The designation of a selling price or WOZ valuation as an outlier is a decision that is independent of other records, and hence, of the sample. In many period - stratum combinations there are enough records without outliers, so that the bootstrap sampling is likely to yield samples with enough regular values for selling price or WOZ valuations so that the SPAR tool that performs the estimation of the SPAR index can process the data without further complications. To be able to do this the bootstrap sampler should have control possibilities for regulating the number of incomplete records in relation to the number of complete records, while at the same time providing sufficient variation in the bootstrap samples that it generates.

To avoid the problems caused by missing data, an option would be to impute the missing values first. In Willenborg and Scholtus (2018) several suggestions are made how this could be done. The simplest of these methods would be to estimate averages of ratios of selling prices and WOZ valuations in for each month m in year j, namely $V^g_{j,m}/W^g_{j,m}$ in the notation of Willenborg and Scholtus (2018). This factor then can be used in later months to estimate missing WOZ valuations when selling prices are known. In this way one would obtain a completed file, which in turn can be fed into the SPAR tool. Because the imputation is carried out on the input file, it is part of the bootstrapping: each bootstrap file should be imputed and the completed file should be used as input for the SPAR tool. The imputation will also contribute its part to the variation in the results, as the presence of missings in the bootstrap samples play their part in adding to the variation of the results.

In Willenborg and Scholtus (2018) various alternatives to the SPAR index have been suggested. There is a reasoning behind the approach: to compare the selling price of a house later in the year with a hypothetical selling price of the same house in January of the same year. The SPAR index also emerges within this reasoning.

It is interesting to investigate some of the indices suggested in Willenborg and Scholtus (2018) and compare the results they yield with those of the SPAR index. This would also hold for the bootstrap results.

References

Davison, A. and D. Hinkley (2009). *Bootstrap Methods and their Application*. Cambridge University Press.

Efron, B. and R. Tibshirani (1994). *An Introduction to the Bootstrap*. Chapman and Hall.

Willenborg, L. and S. Scholtus (2018). The SPAR index and some alternative house price indices. Discussion paper, CBS The Hague.

Appendix

A R code

A.1 Preamble

```
# load necessary R libraries for SPAR tool
library(cbsodataR)
library(reshape2)
library(data.table)
library(plyr)
library(dplyr)
library(shiny)
library(RODBC)
library(foreign)
library(ggplot2)
library(haven)
library(gdata)
library(lubridate)
library(stringr)

# choose settings
# number of bootstrap samples
B <- 500
# stratifying variables
strat <- c('Periode', 'Regio', 'Objecttype', 'Status')
# random seed (set to NULL if no seed is required)
seedwaarde <- NULL
seedwaarde <- 37
# base year (index = 100)
Par_Referentiejaar <- 2012
```

A.2 Bootstrap sampler for SPAR (BSS) - part 1

```
print('Read data and sort file by stratum')
start <- Sys.time()

# read original data
brondata <- read.csv2(file = 'input/Tbl_brondata.csv',
                      header = TRUE, stringsAsFactors = FALSE)

# add Status
brondata$Status <- ifelse(!is.na(brondata$KoopSom),
                          ifelse(!is.na(brondata$WozWaarde),
                                 'regular', 'Woz missing'),
                          ifelse(!is.na(brondata$WozWaarde),
                                 'KoopSom missing', 'both missing'))
table(brondata$Status)
# only 'regular' and 'Woz missing' occur in practice

# order file by stratifying variables
any(is.na(brondata$Periode)) # no NAs
any(is.na(brondata$Regio)) # no NAs
any(is.na(brondata$Objecttype)) # no NAs
any(is.na(brondata$Status)) # no NAs
brondata <- brondata[do.call(order, brondata[ ,strat]), ]

end <- Sys.time()
duur <- end - start
print(duur)

print('Create bootstrap sample indicators')
start <- Sys.time()

draw.boot <- function(data, B) {
  N <- nrow(data)
  data[ , paste0('boot.', 1:B)] <-
    rmultinom(n = B, size = N, prob = rep(1, N))
  return(data)
}

if (!is.null(seedwaarde)) set.seed(seedwaarde)

bootdata <- ddply(.data = brondata,
                  .variables = strat,
                  .fun = draw.boot,
                  B = B)

end <- Sys.time()
duur <- end - start
print(duur)

# save file of bootstrap sample indicators
save(bootdata, file = 'output/Tbl_brondata_allboots.Rda')
```

A.3 Bootstrap sampler for SPAR (BSS) - part 2

Note: This part of the BSS could be split up and run in parallel on different machines.

```
# load file of bootstrap sample indicators
load(file = 'output/Tbl_brondata_allboots.Rda')

print('Create and write bootstrap sample files')
start <- Sys.time()

N <- nrow(bootdata)

for (b in 1:B) {
  cat(sprintf('Starting work on bootstrap file %d\n', b))

  sel <- bootdata[ ,paste0('boot.', b)]

  # select (multiple copies of) rows
  # that have been selected in the bootstrap sample
  sel.boot <- unlist(sapply((1:N)[sel != 0],
                            function(s) rep(s, sel[s])))
  bootfile <- bootdata[sel.boot, -grep('boot', names(bootdata))]

  print(nrow(bootfile) == nrow(bootdata)) # check

  # process bootstrap sample using R code of SPAR tool
  Tbl_brondata <- bootfile
  source('SPAR_bootstrap_SPARtool.R') # run SPAR tool

  # store results
  Tbl_aggregaten[ ,sprintf('SPARmutatie.%d', b)] <-
    Tbl_aggregaten$SPARmutatie
  Tbl_aggregaten[ ,sprintf('Index_HERSCHAALD.%d', b)] <-
    Tbl_aggregaten$Index_HERSCHAALD
  Tbl_aggregaten[ ,sprintf('Groeivoet.%d', b)] <-
    Tbl_aggregaten$Groeivoet

  kolommen <- c(setdiff(strat, 'Status'),
                sprintf('SPARmutatie.%d',b),
                sprintf('Index_HERSCHAALD.%d',b),
                sprintf('Groeivoet.%d', b))
  if (b == 1) {
    res_boot <- Tbl_aggregaten[ , kolommen]
  } else {
    res_boot <- merge(x = res_boot,
                      y = Tbl_aggregaten[ , kolommen],
                      by = c(setdiff(strat, 'Status')), all = TRUE)
  }

} # end of for loop

end <- Sys.time()
duur <- end - start
print(duur)

# save indices based on bootstrap samples
save(res_boot, file = paste0('output/res_boot_',B,'.Rda'))
```

A.4 Bootstrap collector for SPAR (BCS)

```
# load indices based on bootstrap samples
load(file = paste0('output/res_boot_',B,'.Rda'))

# compute indices based on original data
Tbl_brondata <- brondata
source('SPAR_bootstrap_SPARtool.R') # run SPAR tool

res <-
  Tbl_aggregaten[ ,c(setdiff(strat, 'Status'),
                     'Aantal_origineel', 'Aantal_missing',
                     'SPARmutatie', 'Index_HERSCHAALD', 'Groeivoet')]

# compute bootstrap variance and bias
res$Var_SPARmutatie <-
  apply(res_boot[ ,grep('SPARmutatie.', names(res_boot))], 1, var)
res$Var_Index_HERSCHAALD <-
  apply(res_boot[ ,grep('Index_HERSCHAALD.', names(res_boot))], 1, var)
res$Var_Groeivoet <-
  apply(res_boot[ ,grep('Groeivoet.', names(res_boot))], 1, var)
res$Mean_SPARmutatie <-
  apply(res_boot[ ,grep('SPARmutatie.', names(res_boot))], 1, mean)
res$Mean_Index_HERSCHAALD <-
  apply(res_boot[ ,grep('Index_HERSCHAALD.', names(res_boot))], 1, mean)
res$Mean_Groeivoet <-
  apply(res_boot[ ,grep('Groeivoet.', names(res_boot))], 1, mean)
res$Bias_SPARmutatie <-
  res$Mean_SPARmutatie - res$SPARmutatie
res$Bias_Index_HERSCHAALD <-
  res$Mean_Index_HERSCHAALD - res$Index_HERSCHAALD
res$Bias_Groeivoet <-
  res$Mean_Groeivoet - res$Groeivoet

# compute 95% confidence intervals [L95, U95] based on
# bootstrap variances (using normal approximation)
res$L95_SPARmutatie <-
  res$SPARmutatie - qnorm(0.975)*sqrt(res$Var_SPARmutatie)
res$L95_Index_HERSCHAALD <-
  res$Index_HERSCHAALD - qnorm(0.975)*sqrt(res$Var_Index_HERSCHAALD)
res$L95_Groeivoet <-
  res$Groeivoet - qnorm(0.975)*sqrt(res$Var_Groeivoet)
res$U95_SPARmutatie <-
  res$SPARmutatie + qnorm(0.975)*sqrt(res$Var_SPARmutatie)
res$U95_Index_HERSCHAALD <-
  res$Index_HERSCHAALD + qnorm(0.975)*sqrt(res$Var_Index_HERSCHAALD)
res$U95_Groeivoet <-
  res$Groeivoet + qnorm(0.975)*sqrt(res$Var_Groeivoet)
```

```
# compute 95% confidence intervals [L95emp, U95emp] based on
# the empirical bootstrap distribution
res$L95emp_SPARmutatie <-
  apply(res_boot[ ,grep('SPARmutatie.', names(res_boot))], 1,
        quantile, 0.025)
res$U95emp_SPARmutatie <-
  apply(res_boot[ ,grep('SPARmutatie.', names(res_boot))], 1,
        quantile, 0.975)
res$L95emp_Index_HERSCHAALD <-
  apply(res_boot[ ,grep('Index_HERSCHAALD.', names(res_boot))], 1,
        quantile, 0.025)
res$U95emp_Index_HERSCHAALD <-
  apply(res_boot[ ,grep('Index_HERSCHAALD.', names(res_boot))], 1,
        quantile, 0.975)
res$L95emp_Groeivoet <-
  apply(res_boot[ ,grep('Groeivoet.', names(res_boot))], 1,
        quantile, 0.025, na.rm=TRUE)
res$U95emp_Groeivoet <-
  apply(res_boot[ ,grep('Groeivoet.', names(res_boot))], 1,
        quantile, 0.975, na.rm=TRUE)

# save bootstrap results
save(res, file = paste0('output/res_',B,'.Rda'))
```

Colophon

Publisher
Statistics Netherlands
Henri Faasdreef 312, 2492 JP The Hague
www.cbs.nl

Prepress
Statistics Netherlands, Grafimedia

Design
Edenspiekermann

Information
Telephone +31 88 570 70 70, fax +31 70 337 59 94
Via contact form: www.cbs.nl/information