

# Temporal aggregation of (growth) timeseries

*Rob van Harreveld and Anita van der Roest*

*2019-10-23*

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Function aggregate</b>	<b>1</b>
<b>3</b>	<b>Aggregation of growth timeseries</b>	<b>3</b>
<b>4</b>	<b>Aggregation methods for growth timeseries</b>	<b>3</b>
4.1	dif1 method . . . . .	4
4.2	dif1s method . . . . .	5
4.3	rel method . . . . .	5
4.4	pct method . . . . .	6

## 1 Introduction

This vignette discusses temporal aggregation of timeseries, i.e. the conversion of a timeseries to a lower frequency. The vignette is an extension of the vignette “Introduction to package regts”, but can be read independently.

Section 2 describes the standard aggregation function **aggregate**. Section 3 discusses aggregation for growth timeseries with several methods, which are described in detail in section 4.

## 2 Function aggregate

The standard function **aggregate** can be used to convert a timeseries to a lower frequency, for example a quarterly to an annual timeseries, or a monthly to a quarterly timeseries.

An example with a monthly timeseries:

```
> regm <- regts(1:24, start = "2016M1")
> regm
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2016	1	2	3	4	5	6	7	8	9	10	11	12
2017	13	14	15	16	17	18	19	20	21	22	23	24

```
> aggregate(regm, FUN = mean, nfrequency = 4)
```

	Qtr1	Qtr2	Qtr3	Qtr4
2016	2	5	8	11
2017	14	17	20	23

The observations in the timeseries are split into blocks for each quarter and **FUN** is applied to the three observations in that quarter. The result is a quarterly timeseries holding the aggregated values.

Another example with a quarterly timeseries:

```
> regq <- regts(1:10, start = "2016q1")
> regq
```

	Qtr1	Qtr2	Qtr3	Qtr4
2016	1	2	3	4
2017	5	6	7	8
2018	9	10		

```
> aggregate(regq, FUN = sum)
```

	+1
2016 10	26

The quarters in 2016 add up to 10, the quarters in 2017 to 26. Note that the two quarters in 2018 are ignored. The `aggregate` function skips the incomplete years at the end.

Incomplete years at the beginning are also skipped. For example, the next timeseries starts in the second quarter:

```
> regq2 <- regts(2:10, start = "2016q2")
> regq2
```

	Qtr1	Qtr2	Qtr3	Qtr4
2016		2	3	4
2017	5	6	7	8
2018	9	10		

```
> aggregate(regq2, FUN = sum)
```

2017	26
------	----

The quarters in 2017 add up to 26 as before. But now the information for 2016 is also not complete, and therefore ignored.

Thus for `regts` objects the `aggregate` function skips all incomplete years at the beginning and end of the timeseries. However for standard timeseries (`ts`) the implementation of `aggregate` only skips incomplete years at the end. The previous example but now with a `ts` timeseries:

```
> tq2 <- ts(2:10, start = c(2016,2), frequency = 4)
> tq2
```

	Qtr1	Qtr2	Qtr3	Qtr4
2016		2	3	4
2017	5	6	7	8
2018	9	10		

```
> aggregate(tq2, FUN = sum)
```

```
Time Series:
Start = 2016.25
End = 2017.25
Frequency = 1
[1] 14 30
```

The result is an unconventional timeseries with frequency `year` but a shifted period (the start is 2016.25). The result for 2016.25 is the sum of the observations in period 2016Q2/2017Q1.

In all other cases the `aggregate` function works exactly the same for `regts` and `ts` objects.

### 3 Aggregation of growth timeseries

As is shown in the next section the function `aggregate` does not yield correct results for timeseries with absolute, relative or percentage changes. For growth timeseries the function `aggregate_gr` can be used. It accepts both `regts` and `ts` timeseries. The function implements several methods (`dif1`, `dif1s`, `rel` and `pct`) for different types of input timeseries.

Methods `dif1` and `dif1s` assume that the input timeseries is a first difference:

$$x_t = z_t - z_{t-1}$$

where  $t$  is the period (like year or quarter). These methods calculate the low frequency first difference timeseries. For method `dif1s` the result is also scaled. An example for both methods, the results are annual timeseries:

```
> xdif <- diff(regts(1:8, start = "2019q1"))
> xdif
```

	Qtr1	Qtr2	Qtr3	Qtr4
2019		1	1	1
2020	1	1	1	1

```
> aggregate_gr(xdif, method = "dif1")
```

2020 16

```
> aggregate_gr(xdif, method = "dif1s") # result is also scaled
```

2020 4

The `rel` and `pct` methods assume that the input timeseries is a one-period relative or percentage change and calculate the exact low frequency relative or percentage change. The methods are described in detail in the next section.

### 4 Aggregation methods for growth timeseries

This section describes the methods implemented in function `aggregate_gr`, including the derivation of the corresponding mathematical formulas.

Suppose we are temporally aggregating a timeseries  $x$  with  $n$  subperiods to a timeseries  $X$  with a lower frequency. In other words we want to convert a timeseries  $x$  with a high frequency (monthly, quarterly) to a new timeseries with a lower frequency (quarterly, annual).

Let  $x_{t,i}$  stand for the value of  $x$  in subperiod  $i$  of main period  $t$ . The index  $i$  takes on values in the range  $1..n$ . We always interpret  $x_{t,0}$  to mean  $x_{t-1,n}$ . The time index  $t$  refers to the periods in the time domain of  $X$ . Many conversion methods only need the observations subperiod  $i$  of main period  $t$ . For example the `mean` method calculates  $X_t$  as follows

$$X_t = \sum_{i=1}^n x_{t,i}/n$$

However there are transformations which are not so straightforward. A case in point is when  $x$  is a high frequency first difference and must be converted to a low frequency first difference. A simple averaging of  $n$  high frequency observations will not give the correct answer. For example, for a quarterly growth series the yearly aggregates also depend on the quarterly observations of the previous year. Therefore we cannot use function `aggregate`, function `aggregate_gr` must be applied. In the next subsections a more detailed description of the available frequency conversion methods for this function is given.

## 4.1 dif1 method

Define  $x$  to be the first difference of  $z$ . Thus

$$x_{t,i} = z_{t,i} - z_{t,i-1}$$

We can also define  $z_{t,i}$  in terms of  $x$  as follows

$$\begin{aligned} z_{t,i} &= z_{t,i-1} + x_{t,i} \\ &= z_{t-1,n} + \sum_{j=1}^i x_{t,j} \end{aligned}$$

Now define  $Z_t$  as the level timeseries in time domain  $t$  corresponding to  $z$

$$\begin{aligned} Z_t &= \sum_{i=1}^n z_{t,i} \\ &= \sum_{i=1}^n z_{t-1,n} + \sum_{i=1}^n \sum_{j=1}^i x_{t,j} \\ &= \sum_{i=1}^n z_{t-2,n} + \sum_{i=1}^n \sum_{j=1}^n x_{t-1,j} + \sum_{i=1}^n \sum_{j=1}^i x_{t,j} \end{aligned}$$

Then  $X$  in any period  $t$  can be calculated from

$$X_t = Z_t - Z_{t-1}$$

Since  $Z_{t-1}$  can be written as

$$Z_{t-1} = \sum_{i=1}^n z_{t-2,n} + \sum_{i=1}^n \sum_{j=1}^i x_{t-1,j}$$

it is easy to show that  $X_t$  does not depend on  $z_{t-2,n}$ . In any practical algorithm  $z_{t-2,n}$  can be set to 0.

A compact equation for  $X_t$  can be derived by substituting the equations for  $Z_t$  and  $Z_{t-1}$  into the definition of  $X_t$

$$X_t = \sum_{i=1}^n \left( \sum_{j=i+1}^n x_{t-1,j} + \sum_{j=1}^i x_{t,j} \right)$$

This expression can be further simplified by changing the order of summation. For the first summation, we can write

$$\sum_{i=1}^n \sum_{j=i+1}^n x_{t-1,j} = \sum_{j=2}^n \sum_{i=1}^{j-1} x_{t-1,j} = \sum_{j=2}^n (j-1)x_{t-1,j}$$

Similarly,

$$\sum_{i=1}^n \sum_{j=1}^i x_{t,j} = \sum_{j=1}^n \sum_{i=j}^n x_{t,j} = \sum_{j=1}^n (n-j+1)x_{t,j}$$

The final equation is given by

$$X_t = \left( \sum_{j=2}^n (j-1)x_{t-1,j} + \sum_{j=1}^n (n-j+1)x_{t,j} \right)$$

## 4.2 dif1s method

The **dif1s** method is similar to the **dif1** method described in the previous subsection. The only difference is that the aggregate  $Z_t$  is calculated with an average instead of a sum:

$$Z_t = \sum_{i=1}^n z_{t,i} / n$$

Further derivations are analogous. The result for **dif1s** is the result for **dif1** divided by  $n$ :

$$X_t = \left( \sum_{j=2}^n (j-1)x_{t-1,j} + \sum_{j=1}^n (n-j+1)x_{t,j} \right) / n$$

## 4.3 rel method

Define  $x$  to be the relative change in  $z$ . Thus

$$x_{t,i} = (z_{t,i} - z_{t,i-1}) / z_{t,i-1}$$

We can also define  $z_{t,i}$  in terms of  $x$  as follows

$$\begin{aligned} z_{t,i} &= z_{t,i-1}(1 + x_{t,i}) \\ &= z_{t-1,n} \prod_{j=1}^i (1 + x_{t,j}) \\ &= z_{t-2,n} \prod_{j=1}^n (1 + x_{t-1,j}) \prod_{j=1}^i (1 + x_{t,j}) \end{aligned}$$

Now define  $Z_t$  as the level timeseries in time domain  $t$  corresponding to  $z$

$$\begin{aligned} Z_t &= \sum_{i=1}^n z_{t,i} \\ &= z_{t-1,n} \sum_{i=1}^n \prod_{j=1}^i (1 + x_{t,j}) \\ &= z_{t-2,n} \prod_{j=1}^n (1 + x_{t-1,j}) \sum_{i=1}^n \prod_{j=1}^i (1 + x_{t,j}) \end{aligned}$$

Then  $X$  in any period  $t$  can be calculated from

$$X_t = Z_t / Z_{t-1} - 1$$

Using the expression for  $Z_{t-1}$

$$Z_{t-1} = z_{t-2,n} \sum_{i=1}^n \prod_{j=1}^i (1 + x_{t-1,j})$$

we obtain

$$X_t = \frac{\prod_{j=2}^n (1 + x_{t-1,j}) \sum_{i=1}^n \prod_{j=1}^i (1 + x_{t,j})}{1 + \sum_{i=2}^n \prod_{j=2}^i (1 + x_{t-1,j})} - 1$$

#### 4.4 pct method

Define  $x$  to be the percentage change in  $z$ . Thus

$$x_{t,i} = 100(z_{t,i} - z_{t,i-1})/z_{t,i-1}$$

The derivations are analogous to the **rel** method described in the previous subsection.

The result is

$$X_t = 100 \frac{\prod_{j=2}^n (1 + x_{t-1,j}/100) \sum_{i=1}^n \prod_{j=1}^i (1 + x_{t,j}/100)}{1 + \sum_{i=2}^n \prod_{j=2}^i (1 + x_{t-1,j}/100)} - 1$$