Фото [nutraveller](https://pixabay.com/users/nutraveller-5661007/) посредством [Pixabay](https://pixabay.com/) ([лицензия Pixabay](https://pixabay.com/service/license/) )

# Что такое декомпозиция временных рядов и как она работает?

What is time series decomposition and how does it work?

<https://towardsdatascience.com/what-is-time-series-decomposition-and-how-does-it-work-9b67e007ae90>

[Sachin Date](https://sachin-date.medium.com/?source=post_page-----9b67e007ae90--------------------------------)

[21 июня 2020 г. ·](https://towardsdatascience.com/what-is-time-series-decomposition-and-how-does-it-work-9b67e007ae90?source=post_page-----9b67e007ae90--------------------------------)

[Сезонный компонент 2](#_Toc76851305)

[Компонент тренда 3](#_Toc76851306)

[Циклический компонент 4](#_Toc76851307)

[Компонент шума 4](#_Toc76851308)

[Аддитивные и мультипликативные эффекты 4](#_Toc76851309)

[Пошаговая процедура разложения временного ряда на составляющие тренда, сезона и шума с использованием Python 5](#_Toc76851310)

[Разложение временного ряда с использованием библиотеки statsmodels 13](#_Toc76851311)

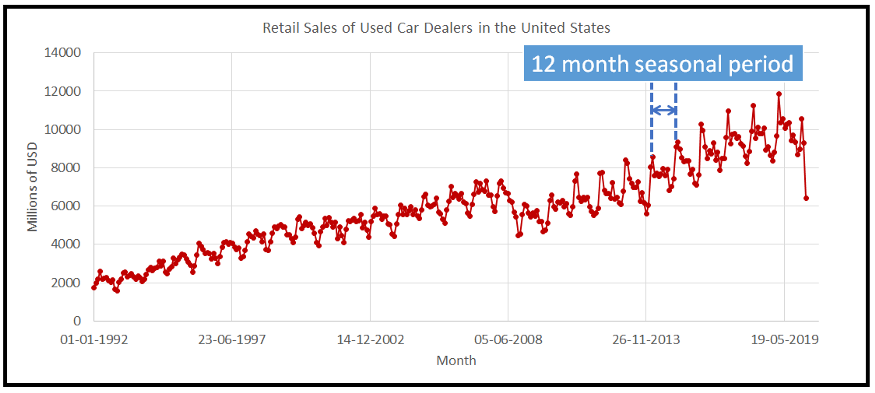
[Ссылки и авторские права 15](#_Toc76851312)

Временной ряд можно рассматривать как состоящий из 4 компонент (составляющих):

* сезон
* тренд
* цикл
* шум

## Сезонный компонент

Сезонный компонент отражает периодические подъемы и спады, встречающиеся во многих наборах данных, например, как на показанном ниже.



Розничные продажи подержанных автомобилей. Источник данных: [US FRED](https://fred.stlouisfed.org/series/MRTSSM44112USN) (Изображение [автора](https://sachin-date.medium.com/))

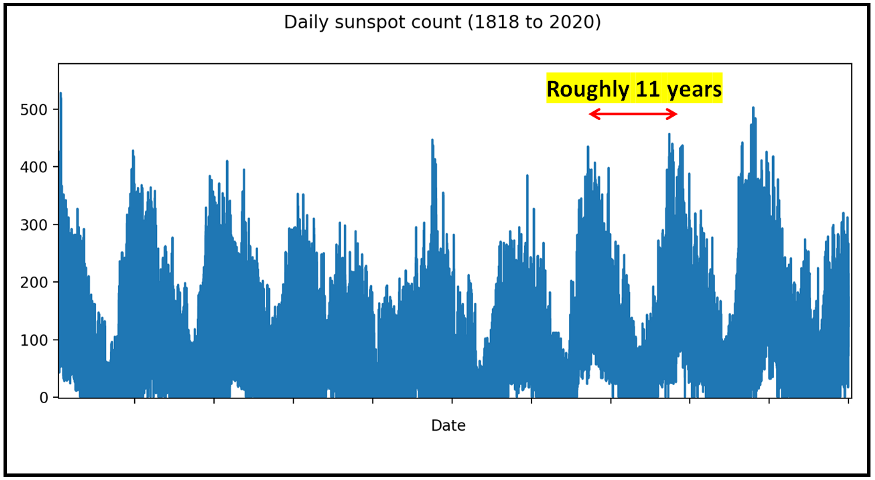
В приведенном примере сезонный период составляет 12 месяцев с пиком продаж в марте и минимумом в ноябре или декабре.

Временной ряд может содержать несколько наложенных друг на друга сезонных периодов. Классическим примером является временной ряд почасовой температуры на метеостанции. Поскольку Земля вращается вокруг своей оси, график часовых температур на метеостанции будет показывать сезонный период в 24 часа. Земля также вращается вокруг Солнца с наклоном относительно последнего, что приводит к сезонным колебаниям температуры. Если вы будете следить за температурой на метеостанции, скажем, в 11 часов утра в течение 365 дней, вы увидите, что появляется вторая модель с периодом в 12 месяцев. 24-часовая дневная компонента накладывается на 12-месячную годовую.

В случае данных о температуре известно, какие физические явления вызывают эти две сезонные закономерности. Но в большинстве случаев невозможно узнать, каковы все факторы, влияющие на сезонность ваших данных. Таким образом, как правило, бывает нелегко выявить все сезонные периоды, находящиеся во временном ряду.

Обычно встречающиеся сезонные периоды – день, неделя, месяц, квартал (или сезон, время года) и год.

Сезонность также наблюдается в гораздо более длительных временных масштабах, например, в солнечном цикле, который составляет около 11 лет.

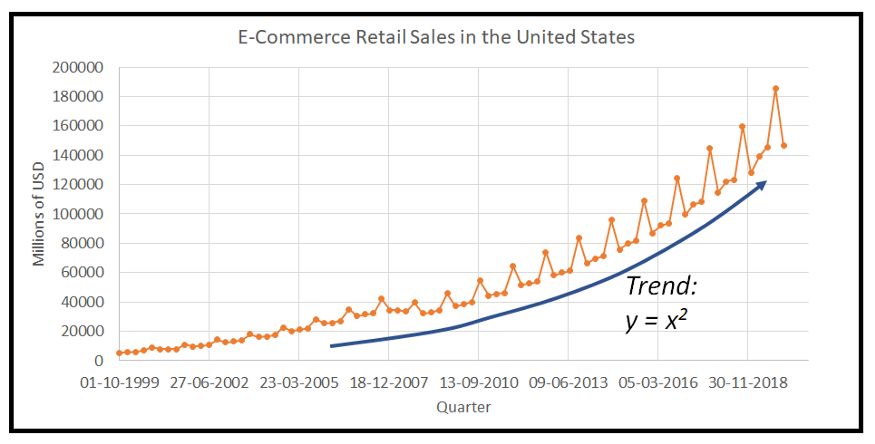


Ежедневный подсчет солнечных пятен. Источник данных: [SILSO](http://www.sidc.be/silso/datafiles) (Изображение [автора](https://sachin-date.medium.com/))

## Компонент тренда

Тренд показывает долгосрочное поведение временного ряда, охватывающее все сезонные периоды.

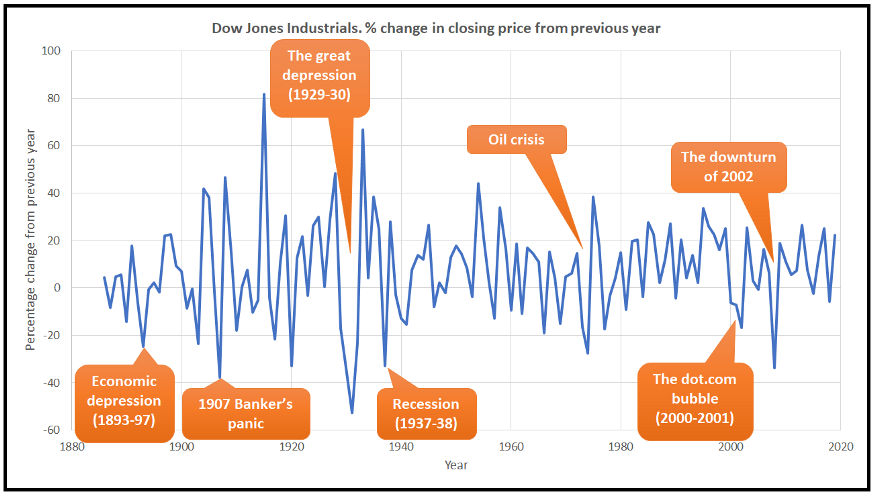
Представленный ниже временной ряд розничных продаж электронной коммерции демонстрируют, по всей видимости, квадратичный тренд (*y = x²*), который охватывает все сезонные периоды.



Розничные продажи электронной коммерции. Источник данных: [US FRED](https://fred.stlouisfed.org/series/ECOMNSA) (Изображение [автора](https://sachin-date.medium.com/))

## Циклический компонент

Циклический компонент представляет явления, как правило, происходящие в пределах нескольких сезонных периодов. У циклических моделей нет фиксированного периода, как у сезонных. Примером циклической модели являются циклы подъема и спада, которые испытывают фондовые рынки в ответ на мировые события.



Dow Jones. Процентное изменение цены закрытия по сравнению с предыдущим годом (1880–2020). Источник данных: [MeasuringWorth.com](https://www.measuringworth.com/datasets/DJA/index.php) через [Википедию](https://en.wikipedia.org/wiki/Dow_Jones_Industrial_Average)) (Изображение [автора](https://sachin-date.medium.com/))

Циклический компонент сложно выделить, его часто «оставляют в покое», считая частью тренда.

## Компонент шума

Шум или случайная составляющая ­– то, что остается, когда вы удаляете сезонность и тренд из временного ряда. Шум – это влияние факторов, о которых вы не знаете или которые невозможно измерить. Это эффект известных неизвестных или неизвестных неизвестных.

## Аддитивные и мультипликативные эффекты

Компоненты тренда, сезонности и шума могут объединяться аддитивным или мультипликативным образами.

**Аддитивная комбинация**  
Если сезонная и шумовая составляющие изменяют тренд на величину, не зависящую от значения тренда, говорят, что тренд, сезонная и шумовая компоненты ведут себя **аддитивным** образом. Это можно представить следующим образом:

*y(i) = t(i) + s(i) + n(i)*

где *y(i)* – значение временного ряда на *i-м* временном шаге.  
*t(i)* – составляющая тренда на *i-м* временном шаге.  
*s(i)* – сезонная составляющая на *i-м* временном шаге.  
*n(i)* – шумовая составляющая на *i-м* временном шаге.

**Мультипликативная комбинация**Если сезонная и шумовая составляющие изменяют тренд на величину, которая зависит от значения тренда, считается, что эти три компонента ведут себя мультипликативным образом, а именно:

*y(i) = t(i) \* s(i) \* n(i)*

## Пошаговая процедура разложения временного ряда на компоненты тренда, сезона и шума с использованием Python.

Существует множество методов декомпозиции (разложения) временного ряда, от простых на основе скользящего среднего до мощных, как, например, STL[[1]](#footnote-1).

В Python в библиотеке *statsmodels* есть метод s*easonal\_decompose(),* позволяющий разложить временной ряд на тренд, сезон и шум в одной строке кода.

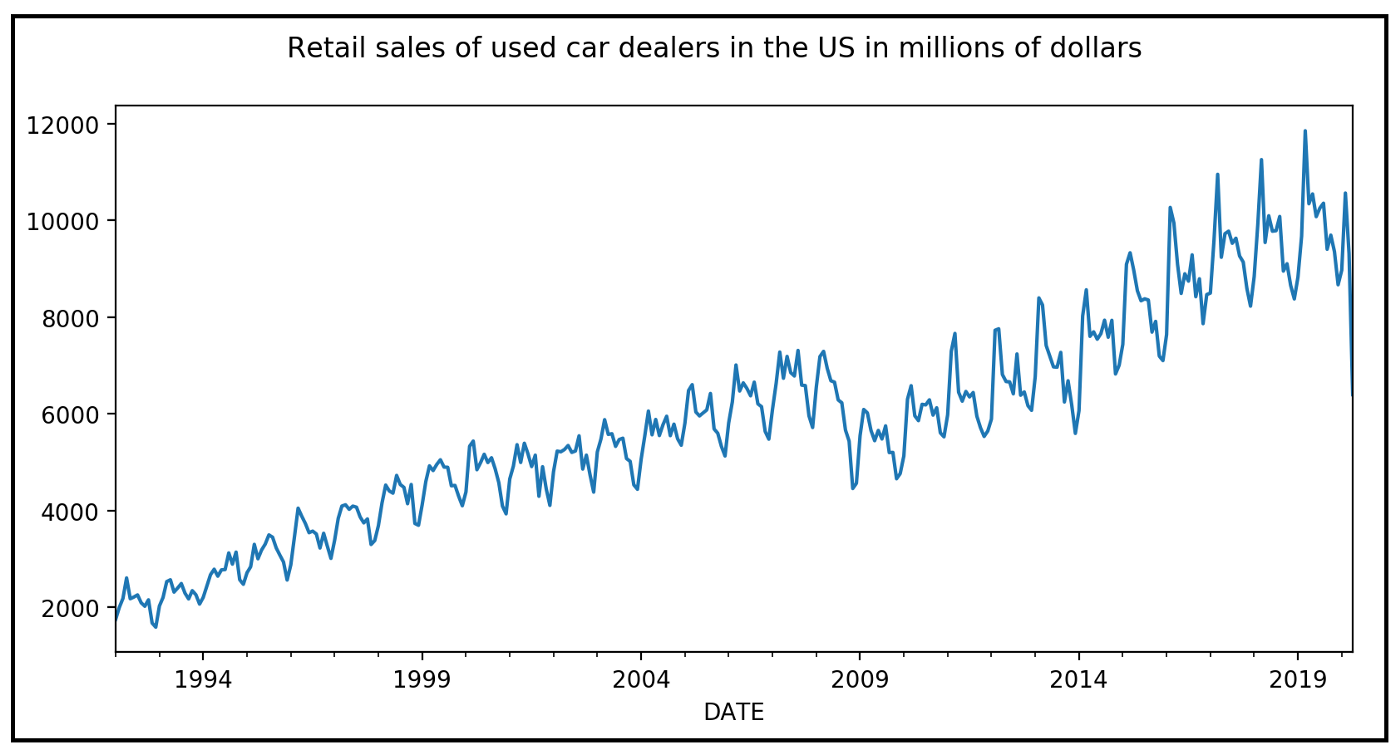
В своих статьях я люблю разбираться в деталях. Прежде чем применять s*easonal\_decompose()*, давайте рассмотрим простую технику декомпозиции временных рядов.

*Разберемся, как работает декомпозиция «под капотом».*

Выполним вручную разложение временного ряда на его составляющие тренда, сезона и шума, используя простую процедуру, основанную на скользящих средних и включающую следующие шаги:

ШАГ 1: определите сезонный период  
ШАГ 2: выделите тренд  
ШАГ 3: выделите сезонность + шум  
ШАГ 4: выделите сезонность  
ШАГ 5: выделите шум

Используем в качестве примера временной ряд розничных продаж дилеров подержанных автомобилей в США.



Розничные продажи подержанных автомобилей. Источник данных: [US FRED](https://fred.stlouisfed.org/series/MRTSSM44112USN) (Изображение [автора](https://sachin-date.medium.com/))

Загрузим данные в DataFrame pandas и построим временной ряд.

**import** pandas **as** pd  
import numpy as np  
import math  
from matplotlib import pyplot as plt

mydateparser = **lambda** x: pd.**datetime**.**strptime**(x, '%d-%m-%y')

df = pd.**read\_csv**('retail\_sales\_used\_car\_dealers\_us\_1992\_2020.csv', **header**=0, **index\_col**=0, **parse\_dates**=['DATE'], **date\_parser**=mydateparser)

fig = plt.figure()fig.suptitle(**'Retail sales of used car dealers in the US in millions of dollars'**)

df[**'Retail\_Sales'**].plot()

plt.show()

Приступим к пошаговой декомпозиции данного ряда.

ШАГ 1. Попробуем угадать периодичность сезонной составляющей. В приведенном примере предполагаем, что это 12 месяцев.

ШАГ 2. Теперь применим 12-месячную [центрированную скользящую среднюю](https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc422.htm). Эта скользящая средняя учитывает 13 месяцев, т.е. по 6 месяцев слева и справа от центрального месяца. 12-месячная центрированная скользящая средняя представляет собой среднее значение двух скользящих средних, которые смещены друг от друга на 1 месяц, фактически являясь взвешенной скользящей средней.

Ниже иллюстрация того, как эта центрированная скользящая средняя может быть рассчитана в Microsoft Excel.

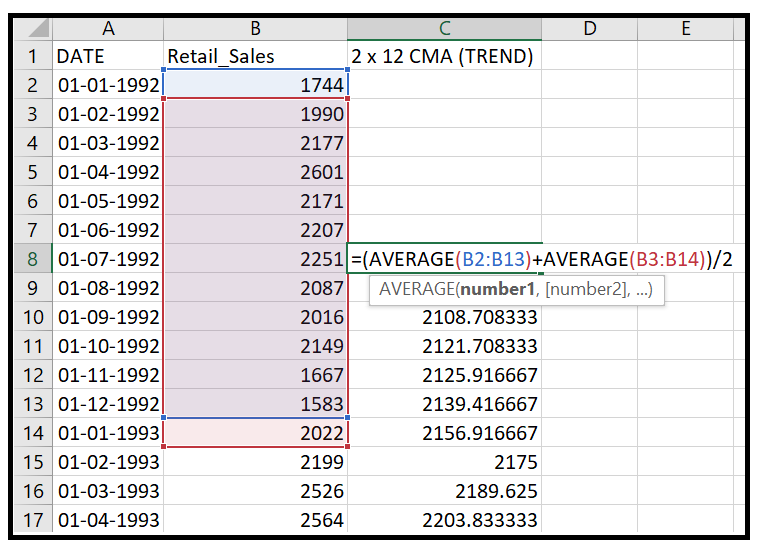


Иллюстрация центрированной скользящей средней 2 x 12 (Изображение [автора](https://sachin-date.medium.com/))

Эта скользящая средняя сгладит сезонность с шумом и выявит тренд.

Вычислим центрированное скользящее среднее в Python.

#Add an empty column to store the 2x12 centered MA values  
df['2 x 12 CMA (TREND)'] = np.nan

#Fill it up with the 2x12 centered MA values  
for i in range(6,df['Retail\_Sales'].size-6):  
 df['2 x 12 CMA (TREND)'][i] = np.round[[2]](#footnote-2)(  
 df['Retail\_Sales'][i - 6] \* 1.0 / 24 +   
 (  
 df['Retail\_Sales'][i - 5] +   
 df['Retail\_Sales'][i - 4] +   
 df['Retail\_Sales'][i - 3] +   
 df['Retail\_Sales'][i - 2] +   
 df['Retail\_Sales'][i - 1] +   
 df['Retail\_Sales'][i] +   
 df['Retail\_Sales'][i + 1] +   
 df['Retail\_Sales']i + 2] +   
 df['Retail\_Sales'][i + 3] +   
 df['Retail\_Sales'][i + 4] +   
 df['Retail\_Sales'][i + 5]  
 ) \* 1.0 / 12 +   
 df['Retail\_Sales'][i + 6] \* 1.0 / 24

Обратите внимание, что значения с индексами [i-6] и [i + 6] берутся с весом 1 / 24, в то время как остальные значения с весом 1 / 12.

Построим полученный временной ряд, значения которого находятся в столбце «2 x 12 CMA (TREND)».

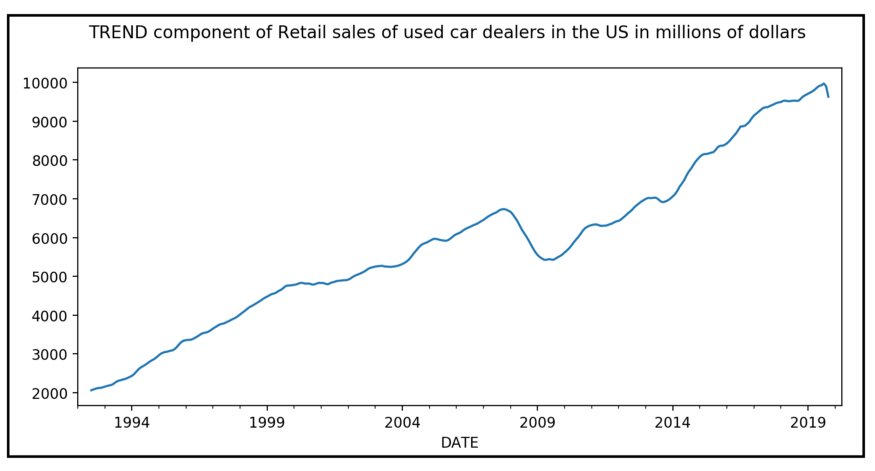
*#plot the trend component*fig = plt.figure()

fig.suptitle('TREND component of Retail sales of used car dealers in the US in millions of dollars')

df['2 x 12 CMA (TREND)'].plot()

plt.show()

Видим, что применение скользящего среднего выявило тренд временного ряда.



(Изображение [автора](https://sachin-date.medium.com/))

ШАГ 3. Теперь нужно принять решение относительно мультипликативности / аддитивности. В зависимости от того, является ли комбинация тренда, сезона и шума мультипликативной или аддитивной, необходимо разделить на компонент тренда или вычесть его из исходного временного ряда, чтобы получить сезонный компонент и компонент шума. Для нашего временного ряда продаж автомобилей видим, что сезонные колебания изменяются пропорционально значению временного ряда. Следовательно, предполагаем, что сезонность мультипликативна. Также предположим, что и шум является мультипликативным.

Таким образом, полагаем, что временной ряд розничных продаж подержанных автомобилей имеет следующую модель мультипликативной декомпозиции:

*значение временного ряда = составляющая тренда \* сезонная составляющая \* шумовая составляющая*

Следовательно:

*сезонная составляющая \* шумовая составляющая = значение временного ряда / составляющая тренда*

Добавим новый столбец в наш фрейм данных и заполним его произведением сезонной и шумовой составляющих, используя приведенную выше формулу.

df['SEASONALITY AND NOISE'] = df['Retail\_Sales']/df['2 x 12 CMA (TREND)']

Построим созданный столбец. Увидим сезонность и шум.

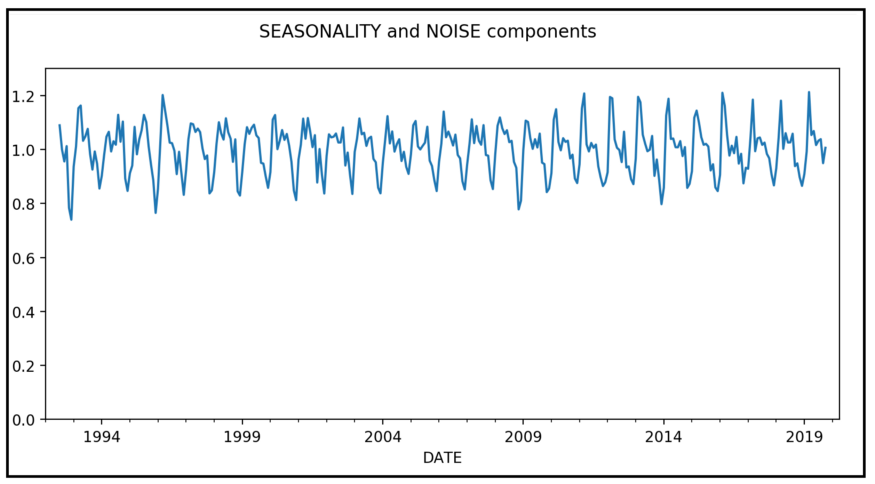
fig = plt.figure()

fig.suptitle('SEASONALITY and NOISE components')

plt.ylim(0, 1.3)

df['SEASONALITY AND NOISE'].plot()

plt.show()



(Изображение [автора](https://sachin-date.medium.com/))

ШАГ 4. Далее выделим сезонную составляющую из смеси сезонности и шума, вычислив среднее значение сезонной составляющей для всех январей, всех февралей и так далее.

*#first add a month column*df['MONTH'] = df.index.strftime('%m').astype(np.int)

*#initialize the month based dictionaries to store the running total of the month wise seasonal sums and counts*average\_seasonal\_values = {1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0}

average\_seasonal\_value\_counts = {1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0}

*#calculate the sums and counts*for i in range(0, df['SEASONALITY AND NOISE'].size):  
 if math.isnan(df['SEASONALITY AND NOISE'][i]) is False:  
 average\_seasonal\_values[df['MONTH'][i]] =   
 average\_seasonal\_values[df['MONTH'][i]] +  
 df['SEASONALITY AND NOISE'][i]

average\_seasonal\_value\_counts[df['MONTH'][i]] =  
 average\_seasonal\_value\_counts[df['MONTH'][i]] + 1

*#calculate the average seasonal component for each month*for i in range(1, 13):  
 average\_seasonal\_values[i] = average\_seasonal\_values[i] / average\_seasonal\_value\_counts[i]

*#create a new column in the data frame and fill it with the value of the average seasonal component for the corresponding month*

df['SEASONALITY'] = np.nan

for i in range(0, df['SEASONALITY AND NOISE'].size):  
 if math.isnan(df['SEASONALITY AND NOISE'][i]) is False:  
 df['SEASONALITY'][i] =   
 average\_seasonal\_values[df['MONTH'][i]]

Построим график сезонной компоненты.

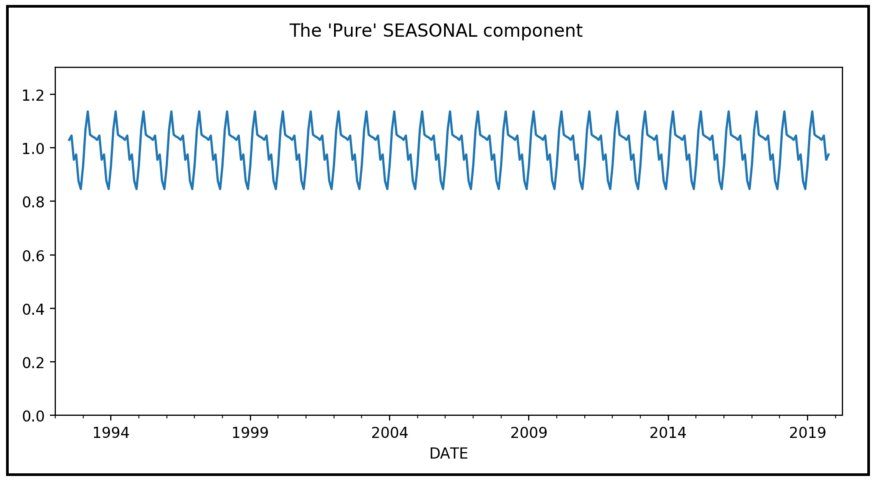
*plot the seasonal component*fig = plt.figure()

fig.suptitle('The \'pure\' SEASONAL component')

plt.ylim(0, 1.3)

df['SEASONALITY'].plot()

plt.show()



(Изображение [автора](https://sachin-date.medium.com/))

ШАГ 5. Наконец, разделим полученную ранее смесь шума и сезона на усредненное сезонное значение, чтобы получить шумовой компонент для каждого месяца.

*шумовая составляющая = сезонная составляющая \* шумовая составляющая / усредненная сезонная составляющая*

df['NOISE'] = df['SEASONALITY AND NOISE']/df['SEASONALITY']

*#plot the seasonal component*

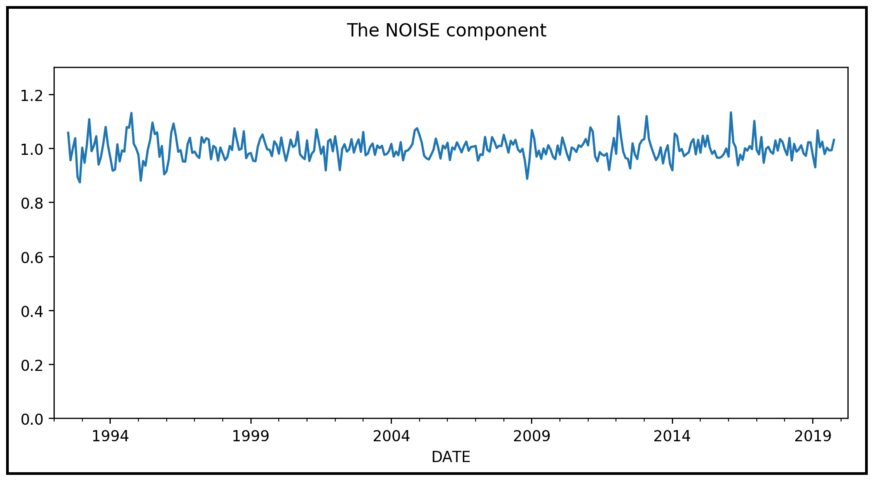
ig = plt.figure()

fig.suptitle('The NOISE component')

plt.ylim(0, 1.3)

df['NOISE'].plot()

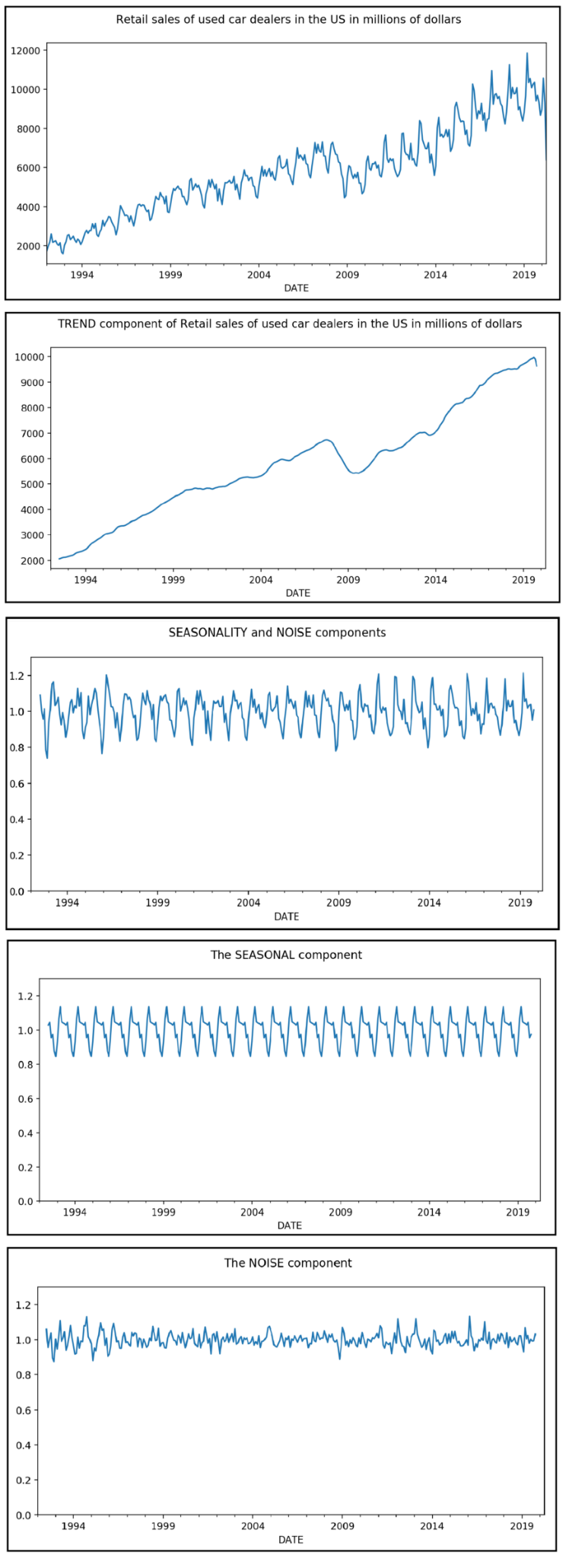
plt.show()



(Изображение [автора](https://sachin-date.medium.com/))

Что же мы сделали? Просто вручную выполнили процедуру разложения временного ряда на его трендовую, сезонную и шумовую составляющие.

Вот картинки временного ряда и его компонент.



(Изображение [автора](https://sachin-date.medium.com/))

## Разложение временного ряда с использованием библиотеки statsmodels

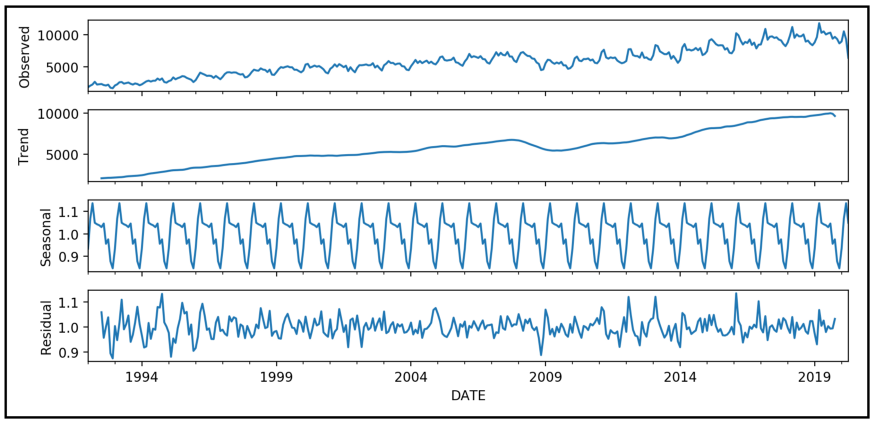
Теперь, когда мы знаем, как декомпозиция работает изнутри, мы можем осознанно применять season\_decompose() из statsmodels, чтобы выполнить всю вышеперечисленную процедуру в одной строке кода.

from statsmodels.tsa.seasonal import seasonal\_decompose

components = seasonal\_decompose(df['Retail\_Sales'], model='multiplicative')

components.plot()

Вновь картинки временного ряда и его компонент.



Результат season\_decompose() для набора данных о розничных продажах подержанных автомобилей (изображение [автора](https://sachin-date.medium.com/))

Полный код Python:

import pandas as pd

import numpy as np

import math

from matplotlib import pyplot as plt

#construct the date parser

mydateparser = lambda x: pd.datetime.strptime(x, '%d-%m-%y')

#load the data set into a pandas data frame

df = pd.read\_csv('retail\_sales\_used\_car\_dealers\_us\_1992\_2020.csv', header=0, index\_col=0, parse\_dates=['DATE'], date\_parser=mydateparser)

#plot the data set

fig = plt.figure()

fig.suptitle('Retail sales of used car dealers in the US in millions of dollars')

df['Retail\_Sales'].plot()

plt.show()

#add a column containing a 2 x 12 centered moving average. this column will capture the trend component in the time series

df['2 x 12 CMA (TREND)'] = np.nan

for i in range(6,df['Retail\_Sales'].size-6):

df['2 x 12 CMA (TREND)'][i] = df['Retail\_Sales'][i - 6] \* 1.0 / 24 + ( df['Retail\_Sales'][i - 5] +

df['Retail\_Sales'][i - 4] + df['Retail\_Sales'][i - 3] + df['Retail\_Sales'][i - 2] +

df['Retail\_Sales'][i - 1] + df['Retail\_Sales'][i] + df['Retail\_Sales'][i + 1] +

df['Retail\_Sales'][i + 2] + df['Retail\_Sales'][i + 3] + df['Retail\_Sales'][i + 4] +

df['Retail\_Sales'][i + 5]) \* 1.0 / 12 + df['Retail\_Sales'][i + 6] \* 1.0 / 24

#plot the trend component

fig = plt.figure()

fig.suptitle('TREND component of Retail sales of used car dealers in the US in millions of dollars')

df['2 x 12 CMA (TREND)'].plot()

plt.show()

df['SEASONALITY AND NOISE'] = df['Retail\_Sales']/df['2 x 12 CMA (TREND)']

#plot the seasonality and noise components

fig = plt.figure()

fig.suptitle('SEASONALITY and NOISE components')

plt.ylim(0, 1.3)

df['SEASONALITY AND NOISE'].plot()

plt.show()

#calculate the average seasonal component for each month

#first add a month column

df['MONTH'] = df.index.strftime('%m').astype(np.int)

#initialize the month based dictionaries to store the running total of themonth wise seasonal sums and counts

average\_seasonal\_values = {1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0}

average\_seasonal\_value\_counts = {1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0}

#calculate the sums and counts

for i in range(0, df['SEASONALITY AND NOISE'].size):

if math.isnan(df['SEASONALITY AND NOISE'][i]) is False:

average\_seasonal\_values[df['MONTH'][i]] = average\_seasonal\_values[df['MONTH'][i]] +

df['SEASONALITY AND NOISE'][i]

average\_seasonal\_value\_counts[df['MONTH'][i]] =

average\_seasonal\_value\_counts[df['MONTH'][i]] + 1

#calculate the average seasonal component for each month

for i in range(1, 13):

average\_seasonal\_values[i] = average\_seasonal\_values[i] / average\_seasonal\_value\_counts[i]

#create a new column in the data frame and fill it with the value of the average seasonal component for the corresponding month

df['SEASONALITY'] = np.nan

for i in range(0, df['SEASONALITY AND NOISE'].size):

if math.isnan(df['SEASONALITY AND NOISE'][i]) is False:

df['SEASONALITY'][i] = average\_seasonal\_values[df['MONTH'][i]]

#plot the seasonal component

fig = plt.figure()

fig.suptitle('The \'Pure\' SEASONAL component')

plt.ylim(0, 1.3)

df['SEASONALITY'].plot()

plt.show()

df['NOISE'] = df['SEASONALITY AND NOISE']/df['SEASONALITY']

#plot the seasonal [[3]](#footnote-3)component

fig = plt.figure()

fig.suptitle('The NOISE component')

plt.ylim(0, 1.3)

df['NOISE'].plot()

plt.show()

#Do all of the above using one line of code!

from statsmodels.tsa.seasonal import seasonal\_decompose

components = seasonal\_decompose(df['Retail\_Sales'], model='multiplicative')

components.plot()

plt.show()

[Ссылка](https://gist.github.com/sachinsdate/40b07da727f8d3d2476dbef4a2f06b6a) на набор данных, использованный в примере.

## Ссылки и авторские права

U.S. Census Bureau, Retail Sales: Used Car Dealers [MRTSSM44112USN], retrieved from FRED, Federal Reserve Bank of St. Louis; <https://fred.stlouisfed.org/series/MRTSSM44112USN>, June 17, 2020, under [FRED copyright terms](https://fred.stlouisfed.org/legal/).

U.S. Census Bureau, E-Commerce Retail Sales [ECOMNSA], retrieved from FRED, Federal Reserve Bank of St. Louis; <https://fred.stlouisfed.org/series/ECOMN>, under [FRED copyright terms](https://fred.stlouisfed.org/legal/).

SILSO, World Data Center — Sunspot Number and Long-term Solar Observations, Royal Observatory of Belgium, on-line Sunspot Number catalogue: <http://www.sidc.be/SILSO/,> 1818–2020 ([CC-BY-NA](https://creativecommons.org/licenses/by-nc/4.0/))

Samuel H. Williamson, “Daily Closing Values of the DJA in the United States, 1885 to Present,” MeasuringWorth, 2020  
URL: <http://www.measuringworth.com/DJA/>

Все изображения в этой статье являются собственностью [Sachin Date в](https://www.linkedin.com/in/sachindate/) соответствии с [CC-BY-NC-SA](https://creativecommons.org/licenses/by-nc-sa/4.0/) , если под изображением не указаны другой источник и авторские права.

*Спасибо за прочтение! Если вам понравилась эта статья,* [***подпишитесь на меня,***](https://sachin-date.medium.com/) *чтобы получать подсказки, практические рекомендации и советы по программированию на тему анализа временных рядов.*

[Sachin Date](https://sachin-date.medium.com/?source=post_sidebar--------------------------post_sidebar-----------)

1. STL - Seasonal-Trend decomposition using LOESS, декомпозиция временного ряда с использованием метода локальных полиномиальных регрессий (LOcal regrESSions — «LOESS» или LOcally WEighted Scatterplot Smoother — «LOWESS») [↑](#footnote-ref-1)
2. np.round отсутствует в коде в конце статьи [↑](#footnote-ref-2)
3. Опечатка. Должно быть “noise” [↑](#footnote-ref-3)