**Removing Trend Information from TS Data**

**Trends in Time Series**

A trend is a long-term increase or decrease in the level of the time series.

*In general, a systematic change in a time series that does not appear to be periodic is known as a trend.*

— Page 5, [Introductory Time Series with R](http://www.amazon.com/dp/0387886974?tag=inspiredalgor-20)

Identifying and understanding trend information can aid in improving model performance; below are a few reasons:

* **Faster Modeling**: Perhaps the knowledge of a trend or lack of a trend can suggest methods and make model selection and evaluation more efficient.
* **Simpler Problem**: Perhaps we can correct or remove the trend to simplify modeling and improve model performance.
* **More Data**: Perhaps we can use trend information, directly or as a summary, to provide additional information to the model and improve model performance.

### Types of Trends

There are all kinds of trends.

Two general classes that we may think about are:

* **Deterministic Trends**: These are trends that consistently increase or decrease.
* **Stochastic Trends**: These are trends that increase and decrease inconsistently.

In general, deterministic trends are easier to identify and remove.

We can think about trends in terms of their scope of observations.

* **Global Trends**: These are trends that apply to the whole time series.
* **Local Trends**: These are trends that apply to parts or subsequences of a time series.

Generally, global trends are easier to identify and address.

### Identifying a Trend

You can plot time series data to see if a trend is obvious or not.

The difficulty is that in practice, identifying a trend in a time series can be a subjective process. As such, extracting or removing it from the time series can be just as subjective.

Create line plots of your data and inspect the plots for obvious trends.

Add linear and nonlinear trend lines to your plots and see if a trend is obvious.

### Removing a Trend

A time series with a trend is called non-stationary.

An identified trend can be modeled. Once modeled, it can be removed from the time series dataset. This is called detrending the time series.

If a dataset does not have a trend or we successfully remove the trend, the dataset is said to be trend stationary.

### Using Time Series Trends in Machine Learning

From a machine learning perspective, a trend in your data represents two opportunities:

1. **Remove Information**: To remove systematic information that distorts the relationship between input and output variables.
2. **Add Information**: To add systematic information to improve the relationship between input and output variables.

Specifically, a trend can be removed from your time series data (and data in the future) as a data preparation and cleaning exercise. This is common when using statistical methods for time series forecasting, but does not always improve results when using machine learning models.

Alternately, a trend can be added, either directly or as a summary, as a new input variable to the supervised learning problem to predict the output variable.

One or both approaches may be relevant for your time series forecasting problem and may be worth investigating.

# How to Use and Remove Trend Information from Time Series Data in Python

by [**Jason Brownlee**](https://machinelearningmastery.com/author/jasonb/) on December 21, 2016 in [**Time Series**](https://machinelearningmastery.com/category/time-series/)

Tweet  **Share** **Share**

Last Updated on August 21, 2019

Our time series dataset may contain a trend.

A trend is a continued increase or decrease in the series over time. There can be benefit in identifying, modeling, and even removing trend information from your time series dataset.

In this tutorial, you will discover how to model and remove trend information from time series data in Python.

After completing this tutorial, you will know:

* The importance and types of trends that may exist in time series and how to identify them.
* How to use a simple differencing method to remove a trend.
* How to model a linear trend and remove it from a sales time series dataset.

Discover how to prepare and visualize time series data and develop autoregressive forecasting models [in my new book](https://machinelearningmastery.com/introduction-to-time-series-forecasting-with-python/), with 28 step-by-step tutorials, and full python code.

Let’s get started.

* **Updated Apr/2019**: Updated the link to dataset.



How to Use and Remove Trend Information from Time Series Data in Python  
Photo by [john78727](https://www.flickr.com/photos/john78727/16611452336/), some rights reserved.

## Trends in Time Series

A trend is a long-term increase or decrease in the level of the time series.

*In general, a systematic change in a time series that does not appear to be periodic is known as a trend.*

— Page 5, [Introductory Time Series with R](http://www.amazon.com/dp/0387886974?tag=inspiredalgor-20)

Identifying and understanding trend information can aid in improving model performance; below are a few reasons:

* **Faster Modeling**: Perhaps the knowledge of a trend or lack of a trend can suggest methods and make model selection and evaluation more efficient.
* **Simpler Problem**: Perhaps we can correct or remove the trend to simplify modeling and improve model performance.
* **More Data**: Perhaps we can use trend information, directly or as a summary, to provide additional information to the model and improve model performance.

### Types of Trends

There are all kinds of trends.

Two general classes that we may think about are:

* **Deterministic Trends**: These are trends that consistently increase or decrease.
* **Stochastic Trends**: These are trends that increase and decrease inconsistently.

In general, deterministic trends are easier to identify and remove, but the methods discussed in this tutorial can still be useful for stochastic trends.

We can think about trends in terms of their scope of observations.

* **Global Trends**: These are trends that apply to the whole time series.
* **Local Trends**: These are trends that apply to parts or subsequences of a time series.

Generally, global trends are easier to identify and address.

### Identifying a Trend

You can plot time series data to see if a trend is obvious or not.

The difficulty is that in practice, identifying a trend in a time series can be a subjective process. As such, extracting or removing it from the time series can be just as subjective.

Create line plots of your data and inspect the plots for obvious trends.

Add linear and nonlinear trend lines to your plots and see if a trend is obvious.

### Removing a Trend

A time series with a trend is called non-stationary.

An identified trend can be modeled. Once modeled, it can be removed from the time series dataset. This is called detrending the time series.

If a dataset does not have a trend or we successfully remove the trend, the dataset is said to be trend stationary.

### Using Time Series Trends in Machine Learning

From a machine learning perspective, a trend in your data represents two opportunities:

1. **Remove Information**: To remove systematic information that distorts the relationship between input and output variables.
2. **Add Information**: To add systematic information to improve the relationship between input and output variables.

Specifically, a trend can be removed from your time series data (and data in the future) as a data preparation and cleaning exercise. This is common when using statistical methods for time series forecasting, but does not always improve results when using machine learning models.

Alternately, a trend can be added, either directly or as a summary, as a new input variable to the supervised learning problem to predict the output variable.

One or both approaches may be relevant for your time series forecasting problem and may be worth investigating.

Next, let’s take a look at a dataset that has a trend.

**Shampoo Sales Dataset**

This dataset describes the monthly number of sales of shampoo over a 3 year period.

The units are a sales count and there are 36 observations. The original dataset is credited to Makridakis, Wheelwright, and Hyndman (1998).

"Month","Sales"

"1-01",266.0

"1-02",145.9

"1-03",183.1

"1-04",119.3

"1-05",180.3

Below is a plot of the entire dataset, where you can learn more and download the dataset.

The dataset shows an increasing trend.



Shampoo Sales Dataset

## Load the Shampoo Sales Dataset

from pandas import read\_csv

from pandas import datetime

from matplotlib import pyplot

def parser(x):

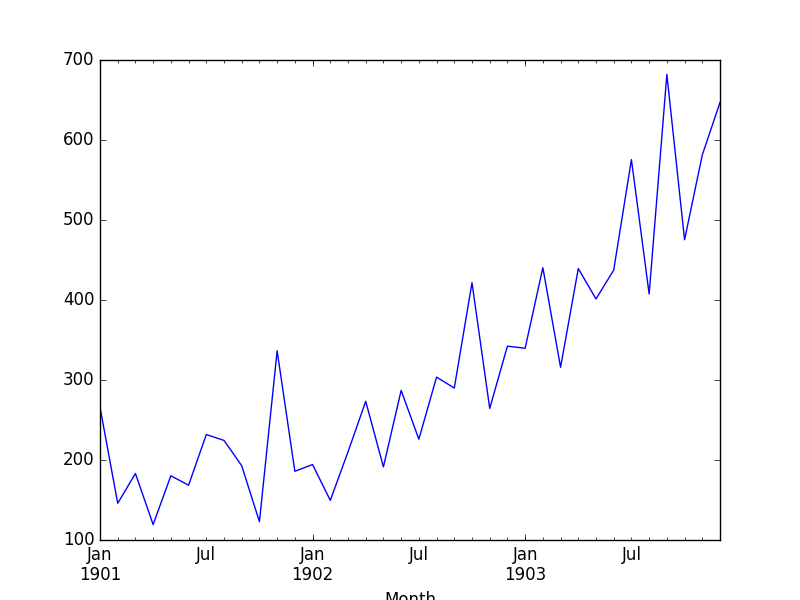
return datetime.strptime('190'+x, '%Y-%m')

series = read\_csv('shampoo-sales.csv', header=0, parse\_dates=[0], index\_col=0, squeeze=True, date\_parser=parser)

series.plot()

pyplot.show()

Running the example loads the dataset and creates a plot.



Shampoo Sales Dataset Plot

## Detrend by Differencing

Perhaps the simplest method to detrend a time series is by differencing.

Specifically, a new series is constructed where the value at the current time step is calculated as the difference between the original observation and the observation at the previous time step.

**value(t) = observation(t) - observation(t-1)**

This has the effect of removing a trend from a time series dataset.

We can create a new difference dataset in Python by implementing this directly. A new list of observations can be created.

Below is an example that creates the difference detrended version of the Shampoo Sales dataset.

from pandas import read\_csv

from pandas import datetime

from matplotlib import pyplot

def parser(x):

return datetime.strptime('190'+x, '%Y-%m')

series = read\_csv('shampoo-sales.csv', header=0, parse\_dates=[0], index\_col=0, squeeze=True, date\_parser=parser)

X = series.values

diff = list()

for i in range(1, len(X)):

value = X[i] - X[i - 1]

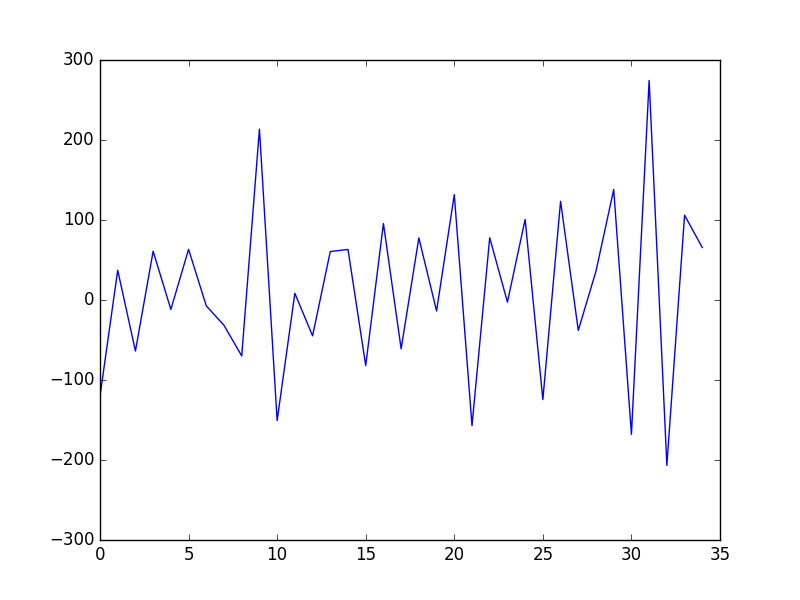
diff.append(value)

pyplot.plot(diff)

pyplot.show()

Running the example creates the new detrended dataset and then plots the time series.

Because no difference value can be created for the first observation (there is nothing for it to be subtracted from), the new dataset contains one less record. We can see that indeed the trend does appear to have been removed.



This approach works well for data with a linear trend. If the trend is quadratic (the change in the trend also increases or decreases), then a difference of the already-differenced dataset can be taken, a second level of differencing. This process can be further repeated if needed.

Because differencing only requires the observation at the previous time step, it can easily be applied to unseen out-of-sample data to either preprocess or provide an additional input for supervised learning.

Next, we will look at fitting a model to describe the trend.

A trend is often easily visualized as a line through the observations.

Linear trends can be summarized by a linear model, and nonlinear trends may be best summarized using a polynomial or other curve-fitting method.

Because of the subjective and domain-specific nature of identifying trends, this approach can help to identify whether a trend is present. Even fitting a linear model to a trend that is clearly super-linear or exponential can be helpful.

In addition to being used as a trend identification tool, these fit models can also be used to detrend a time series.

For example, a linear model can be fit on the time index to predict the observation. This dataset would look as follows:

The predictions from this model will form a straight line that can be taken as the trend line for the dataset. These predictions can also be subtracted from the original time series to provide a detrended version of the dataset.

value(t) = observation(t) - prediction(t)

The residuals from the fit of the model are a detrended form of the dataset. Polynomial curve fitting and other nonlinear models can also be used.

Using scikit-learn LinearRegression Model

from pandas import read\_csv

from pandas import datetime

from sklearn.linear\_model import LinearRegression

from matplotlib import pyplot

import numpy

def parser(x):

return datetime.strptime('190'+x, '%Y-%m')

series = read\_csv('shampoo-sales.csv', header=0, parse\_dates=[0], index\_col=0, squeeze=True, date\_parser=parser)

# fit linear model

X = [i for i in range(0, len(series))]

X = numpy.reshape(X, (len(X), 1))

y = series.values

model = LinearRegression()

model.fit(X, y)

# calculate trend

trend = model.predict(X)

# plot trend

pyplot.plot(y)

pyplot.plot(trend)

pyplot.show()

# detrend

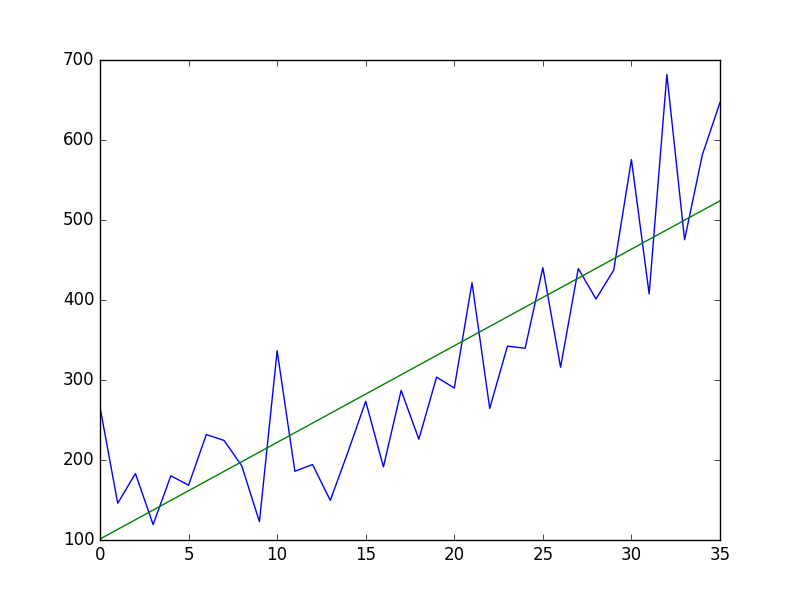
detrended = [y[i]-trend[i] for i in range(0, len(series))]

# plot detrended

pyplot.plot(detrended)

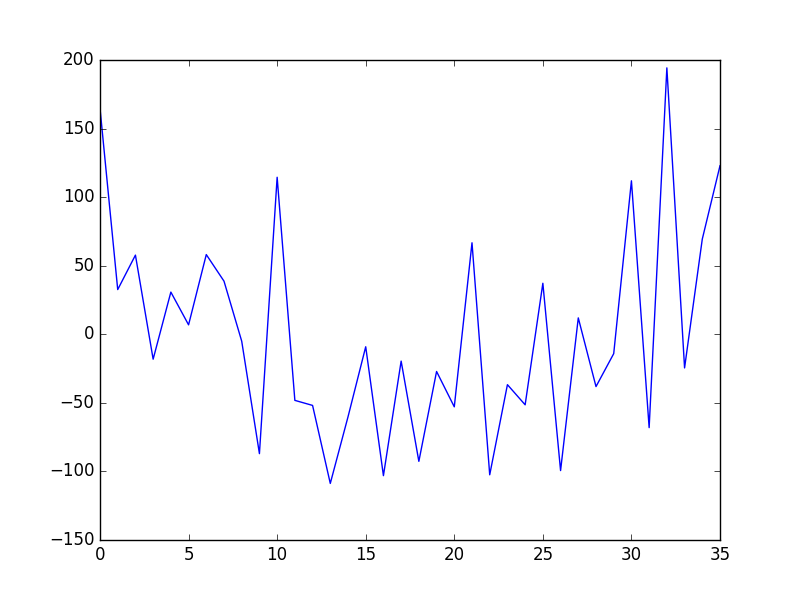
pyplot.show()

Running the example first fits the linear model to the integer-indexed observations and plots the trend line (green) over the original dataset (blue).



Shampoo Sales Dataset Plot With Trend

Next, the trend is subtracted from the original dataset and the resulting detrended dataset is plotted.



Shampoo Sales Dataset Model Detrended

Again, we can see that this approach has effectively detrended the dataset. There may be a parabola in the residuals, suggesting that perhaps a polynomial fit may have done a better job.

Because the trend model takes only the integer index of the observation as input, it can be used on new data to either detrend or provide a new input variable for the model.