



Definitions & Best Techniques for 2025

In this technical paper, InfluxData CTO, Paul Dix, will walk you through what time series is (and isn't), what makes it different from stream processing, full-text search, and other solutions, and how to deliver real-time analytics.



Download Full Tech Paper



Table of Contents

[Time series data and analysis](#)

[Time series examples](#)

[What is time series analysis?](#)

[Time series forecasting methods](#)

[Time series analysis best practices](#)

[FAQ's](#)

Time series data and analysis

Time series analysis looks at data collected over time. For example, a time series metric could be the amount of inventory sold in a store from one day to the next. Often patterns emerge that can predict and prevent issues. A sudden drop in sales would be expensive for the company, so it would help to understand what events precede and predict this kind of change. Time series data is everywhere. As our world becomes increasingly instrumented, sensors and systems emit a relentless stream of **time series data**.

Examples of time series analysis:

- Electrical activity in the brain
- Rainfall measurements
- Stock prices
- Number of sunspots
- Annual retail sales
- Monthly subscribers
- Heartbeats per minute

InfluxDB: The Basics of Time Series Data



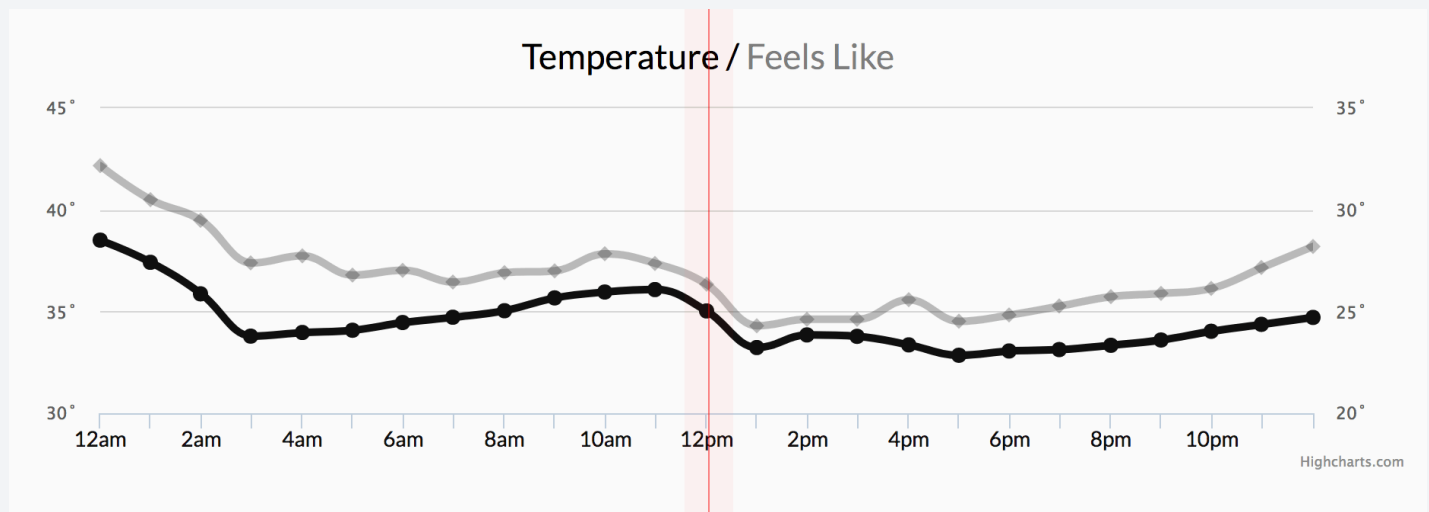
Key time series concepts

- **Time series data** is a collection of data points over time.
- **Time series analysis** is identifying trends, like seasonality, to help forecast a future event.

Time series examples

Weather records, economic indicators and patient health evolution metrics—all are time series data. Time series data could also be server metrics, application performance monitoring, network data, sensor data, events, clicks and many other types of analytics data. The best way to understand time series is to start exploring with some [sample data](#) in [InfluxDB Cloud](#).

Notice how time—depicted at the bottom of the below chart—is the axis.



Example 1: Weather conditions

In the next chart below, note time as the axis over which stock price changes are measured. In investing, a time series tracks the movement of data points, such as a security's price over a specified period of time with data points recorded at regular intervals. This can be tracked over the short term (such as a security's price on the hour over the course of a business day) or the long term (such as a security's price at close on the last day of every month over the course of five years).

Dow Jones Industrial Average (^DJI)

DJI - DJI Real Time Price. Currency in USD

☆ Add to watchlist

24,834.96

+33.60 (+0.14%)

As of 2:56PM EST. Market open.

Summary

Chart

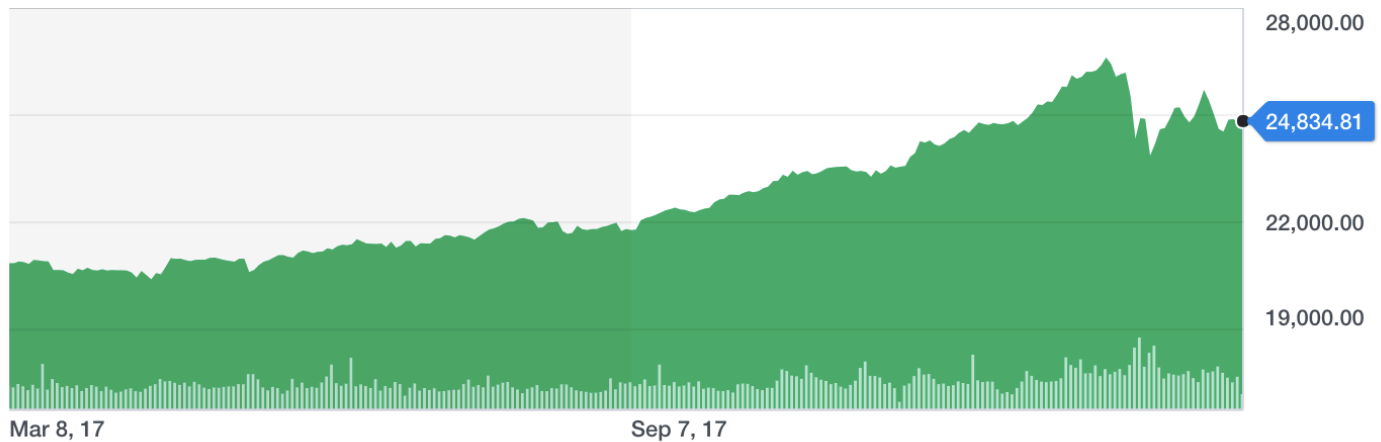
Options

Components

Historical Data

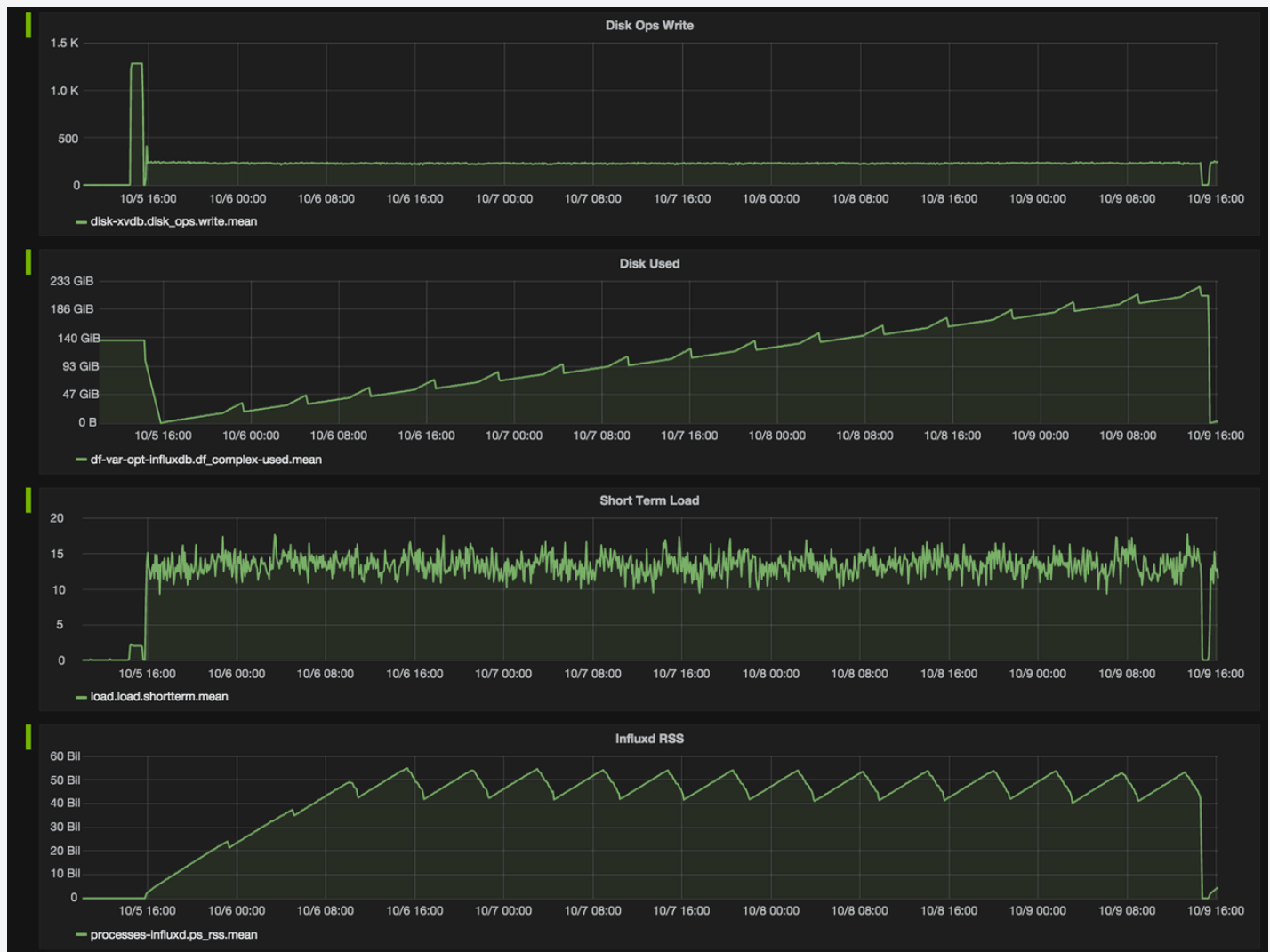
1D 5D 1M 6M YTD 1Y 5Y Max

Full screen



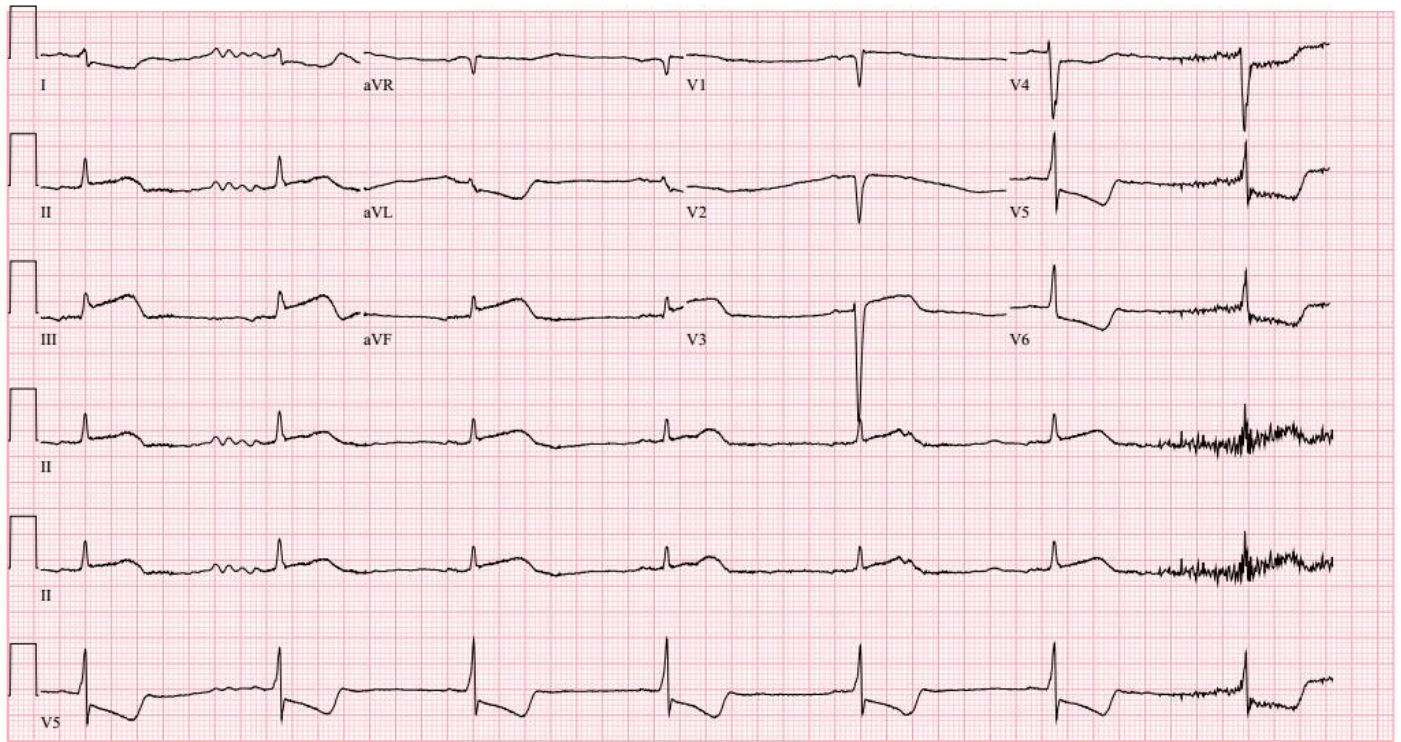
Example 2: Stock exchange

The cluster monitoring example below, depicting disk ops write and usage data, would be familiar to Network Operation Center teams. Remember that monitoring data is time series data.



Example 3: Cluster monitoring

Another familiar example of time series data is patient health monitoring, such as in an electrocardiogram (ECG), which monitors the heart's activity to show whether it is working normally.



Example 4: Health monitoring

In addition to being captured at regular time intervals, time series data can be captured whenever it happens—regardless of the time interval, such as in logs. Logs are a registry of events, processes, messages and communication between software applications and the operating system. Every executable file produces a log file where all activities are noted. Log data is an important contextual source to triage and resolve issues. For example, in networking, an event log helps provide information about network traffic, usage and other conditions.

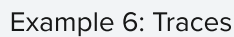

```

Jun 24 13:45:34 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:46:40] "GET /EPA-WASTE/1994/October/Day-00/ HTTP/1.0" 200 330
Jun 24 13:45:36 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:50] "GET /logos/small_gopher.gif HTTP/1.0" 200 935
Jun 24 13:45:38 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:54] "GET /logos/small_ftp.gif HTTP/1.0" 200 124
Jun 24 13:45:40 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:46:55] "GET /docs/EPA-WASTE/1994/October/Day-05 HTTP/1.0" 302 -
Jun 24 13:45:40 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:56] "GET /icons/book.gif HTTP/1.0" 200 156
Jun 24 13:45:41 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:46:56] "GET /EPA-WASTE/1994/October/Day-05/ HTTP/1.0" 200 623
Jun 24 13:45:42 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:58] "GET /logos/us-flag.gif HTTP/1.0" 200 2788
Jun 24 13:45:43 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:12] "GET /docs/EPA-WASTE/1994/October/Day-03 HTTP/1.0" 302 -
Jun 24 13:45:45 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:14] "GET /EPA-WASTE/1994/October/Day-03/ HTTP/1.0" 200 785
Jun 24 13:45:46 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:47:19] "GET /icons/ok2-0.gif HTTP/1.0" 200 231
Jun 24 13:45:48 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:24] "GET /enviro/html/emci/emci_overview.html HTTP/1.0" 200 2352
Jun 24 13:45:49 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:31] "GET /enviro/gif/efacts.gif HTTP/1.0" 200 1367
Jun 24 13:45:50 haproxy epa-http.txt: 202.96.29.111 [30:01:47:34] "GET /PressReleases/ HTTP/1.0" 200 1241
Jun 24 13:45:51 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:37] "GET /enviro/gif/blueball.gif HTTP/1.0" 200 903
Jun 24 13:45:53 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:37] "GET /Rules.html HTTP/1.0" 200 3273
Jun 24 13:45:53 haproxy epa-http.txt: 202.96.29.111 [30:01:47:38] "GET /icons/circle_logo_small.gif HTTP/1.0" 200 2624
Jun 24 13:45:54 haproxy epa-http.txt: 202.96.29.111 [30:01:48:04] "POST /cgi-bin/waisgate/134.67.99.11=earth1.epa.gov=210=/usr1/commwais/indexes/PressReleases=gopher%40earth1.0.00=:free HTTP/1.0" 200 3993
Jun 24 13:45:54 haproxy epa-http.txt: 202.96.29.111 [30:01:48:16] "GET /waisicons/text.xbm HTTP/1.0" 200 527
Jun 24 13:45:55 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:48:22] "GET /Rules.html HTTP/1.0" 200 3273
Jun 24 13:45:57 haproxy epa-http.txt: www-c8.proxy.aol.com [30:01:48:23] "GET /docs/Searchable.html HTTP/1.0" 200 765
Jun 24 13:45:58 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:48:25] "GET /enviro/gif/banner.gif HTTP/1.0" 200 14887
Jun 24 13:54:14 farm-trivia-72 app/web.1: User Load (1.2ms) SELECT "users".* FROM "users" WHERE "users"."id" = $1 ORDER BY "users"."id" ASC LIMIT 1 [["id", 1]]
Jun 24 13:54:14 farm-trivia-72 app/web.1: (1.3ms) SELECT COUNT(*) FROM "products"
Jun 24 13:54:14 farm-trivia-72 heroku/router: at=info method=GET path="/a" host=farm-trivia-72.herokuapp.com request_id=3a095914-087a-4b7a-9f88-81d6e2ba7771 fwd="23.252.53.179" dyno=web.1 connect=1ms service=44ms status=200 bytes=6407
Jun 24 13:54:14 farm-trivia-72 app/web.1: Product Load (1.4ms) SELECT "products".* FROM "products" ORDER BY products.updated_at desc LIMIT 1
Jun 24 13:54:14 farm-trivia-72 app/web.1: User Load (1.4ms) SELECT "users".* FROM "users" ORDER BY users.updated_at desc LIMIT 1
Jun 24 13:54:14 farm-trivia-72 app/web.1: (1.2ms) SELECT COUNT(*) FROM "users"
Jun 24 13:54:14 farm-trivia-72 app/web.1: method=GET path=/a/ format=html controller=rails_admin/main action=dashboard status=200 duration=35.71 view=20.85 db=6.39 remote_ip=23.252.53.179 user_id=1 params={}
Jun 24 13:54:16 farm-trivia-72 heroku/router: at=info method=GET path="/a/product?_pjax=%5Bdata-pjax-container%5D" host=farm-trivia-72.herokuapp.com request_id=4e7f806e-63b2-493a-88d4-ec8ebab5f0a6 fwd="23.252.53.179" dyno=web.1 connect=3ms service=102ms status=200 bytes=17350
Jun 24 13:54:16 farm-trivia-72 app/web.1: Product Load (1.7ms) SELECT "products".* FROM "products" ORDER BY products.id desc LIMIT 20 OFFSET 0
Jun 24 13:54:16 farm-trivia-72 app/web.1: User Load (1.2ms) SELECT "users".* FROM "users" WHERE "users"."id" = $1 ORDER BY "users"."id" ASC LIMIT 1 [["id", 1]]
Jun 24 13:54:16 farm-trivia-72 app/web.1: (1.3ms) SELECT COUNT(*) FROM "products"

```

Example 5: Logs

Traces (a list of the subroutine calls that an application performs during execution) are also time series data. Over the colored bands in the traces chart below, you can see examples of time series data. The goal of tracing is to follow a program's flow and data progression. Tracing encompasses a wide, continuous view of an application to find bugs in a program or application.

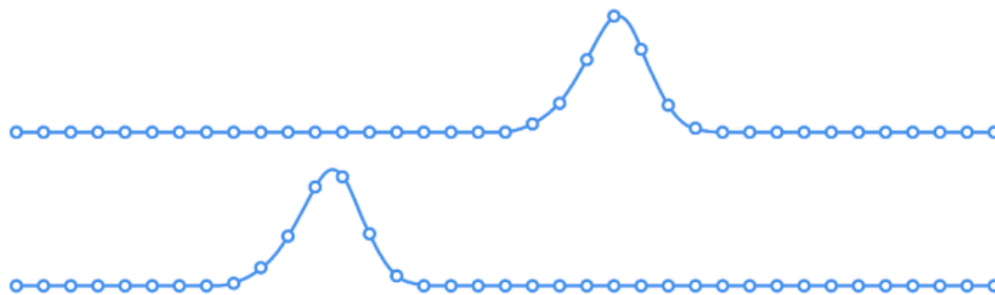


Types of time series data

1. Measurements gathered at regular time intervals (metrics)
2. Measurements gathered at irregular time intervals (events)

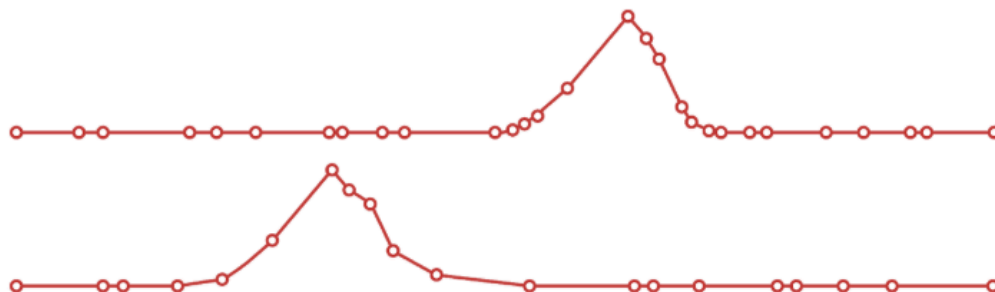
Metrics (Regular)

Measurements gathered at regular time intervals



Events (Irregular)

Measurements gathered at irregular time intervals



In the “Time series data examples” section above:

- Examples 3 (cluster monitoring) and 4 (health monitoring) depict metrics.
- Examples 5 (logs) and 6 (traces) depict events.

Because they happen at irregular intervals, events are unpredictable and cannot be modeled or forecasted since forecasting assumes that whatever happened in the past is a good indicator of what will happen in the future.

A time series data example can be any information sequence that was taken at specific time intervals (whether regular or irregular). Common data examples could be anything from heart rate to the unit price of store goods.

Linear vs. nonlinear time series data

A linear time series is one where, for each data point X_t , that data point can be viewed as a linear combination of past or future values or differences. Nonlinear time series are generated by nonlinear dynamic equations. They have features that cannot be modelled by linear processes: time-changing variance, asymmetric cycles, higher-moment structures, thresholds and breaks. Here are some important considerations when working with linear and nonlinear time series data:

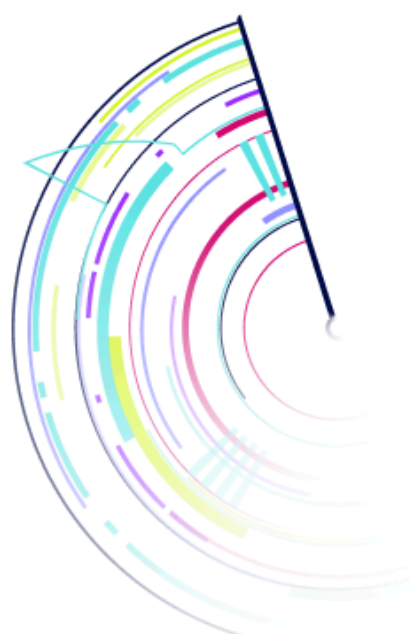
- If a regression equation doesn't follow the rules for a linear model, then it must be a nonlinear model.
- Nonlinear regression can fit an enormous variety of curves.

- The defining characteristic for both types of models are the functional forms.

Time series analysis

Analyzing time series data allows for extracting meaningful statistics and other data characteristics. As the name suggests, time series data is a collection of observations created by repeating measurements over time. Once you have that information, you can plot it on a graph and learn more about precisely what you're tracking.


A very straightforward time series analysis example might be the rise and fall of the temperature over the course of a day. By tracking the specific temperature outside at hourly intervals for 24 hours, you have a complete picture of the rise and fall of the temperature in your area. Then, suppose you know that the next day will be relatively similar in terms of things like precipitation and humidity. In that case, you can make a more educated guess about the temperature at specific times. This analysis is an oversimplified example, yes—but the underlying structure is the same regardless of what it is that you're talking about.



Time Series

A series of data points captured in order over time.

A time series is a collection of data points gathered over a period of time and ordered chronologically. The primary characteristic of a time series is that it's indexed or listed in time order, which is a critical distinction from other types of data sets. If you were to plot the points of time series data on a graph, and one of your axes would always be time.

 influxdata®

Such analysis requires identifying the pattern of an observed time series data set. Once the pattern is established, it can be interpreted, integrated with other data, and used for forecasting (fundamental for machine learning). Machine learning is a type of artificial intelligence that allows computer

programs to actually “learn” and become “smarter” over time, all without being explicitly programmed to do so.

Importance of time series analysis

As more connected devices are implemented and data is expected to be collected and processed in real-time, the ability to handle time series data has become increasingly significant. This will become increasingly critical over the next few years as the Internet of Things, AI, and devices play an ever more important role in all of our lives.

At its core, the Internet of Things is a term used to describe a network of literally billions of connected devices — both creating and sharing data at all times. In a personal context, we’ve already seen this transition begin in “smart” homes across America. Your thermostat knows that when it gets to a certain temperature, it needs to lower the shades in a room to help control the temperature. Or your smart home hub knows that as soon as the last person leaves the house, it is to lock all the doors and turn all the lights off. It wouldn’t be able to get to this point were it not for an interconnected network of sensors exchanging information at all times—making time series analysis all the more critical.

Among other things, time series analysis can effectively - Illustrate that data points taken over time may have some sort of trend or pattern that likely otherwise would have gone undiscovered. - Provide users with a better understanding of the past, thus putting them in a position to better predict the future.

That last point is crucial and is a big part of why time series analysis is used in economics, statistics, and similar fields. Suppose you have historical data for a particular stock, for example, and you know how it has traditionally performed given certain world events. In that case, you can better predict the price when similar events occur in the future. If you know an economic downturn is coming, you can use that insight to make a better and more informed decision about purchasing the stock.

Since the analysis is based on data plotted against time, the first step is to [plot the data](#) and observe any patterns that might occur over time.

Want to learn more? Register for the [Time Series Basics training](#) or [compare](#) options for storing and analyzing time series data.

What is a time series graph?

Time series graphs are simply plots of time series data on one axis (typically Y) against time on the other axis (typically X). Graphs of time series data points can often illustrate trends or patterns in a more accessible, intuitive way.

What are time plot statistics?

A time series plot is a graph in which the x-axis represents some measure of time. In fact, the x-axis is labeled as the time-axis. The y-axis represents the variable being measured. Data points are displayed and connected with straight lines in most cases, allowing for interpretation of the resulting graph.

Time series data can be [visualized in different types of charts](#) to facilitate insight extraction, trend analysis, and anomaly detection. Time series visualization and [dashboarding tools](#) include the InfluxDB UI and [Grafana](#).

The term ‘time series patterns’ describes long-term changes in the series. Whether measured as a trend, seasonal, or cyclic pattern, the correlation can be calculated in a number of ways (linear, exponential, etc.), and the direction may change at any given time.

Time series data is used in time series analysis (historical or real-time) and time series forecasting to detect and predict patterns — essentially looking at change over time. Following is a brief overview of each.

Time series analysis methods

Time series analysis is a method of analyzing a series of data points collected over a period of time. In time series analysis, data points are recorded at regular intervals over a set period of time, rather than intermittently or at random.

Time series analysis is the use of statistical methods to analyze time series data and extract meaningful statistics and characteristics about the data. TSA helps identify trends, cycles, and seasonal variances to aid in the forecasting of a future event. Factors relevant to TSA include stationarity, seasonality and autocorrelation.

Time series analysis can be useful to see how a given variable changes over time (while time itself, in time series data, is often the independent variable). Time series analysis can also be used to examine how the changes associated with the chosen data point compare to shifts in other variables over the same time period.

Time series forecasting methods

Time series forecasting uses information regarding historical values and associated patterns to predict future activity.

Time series forecasting methods include:

- Trend analysis
- Cyclical fluctuation analysis
- Seasonal pattern analysis

As with all forecasting methods, success is not guaranteed. Machine learning is often used for this purpose. So are its classical predecessors: [Error, Trend, Seasonality Forecast \(ETS\)](#), [Autoregressive Integrated Moving Average \(ARIMA\)](#) and [Holt-Winters](#).

To 'see things' ahead of time, time series modeling (a forecasting method based on time series data) involves working on time-based data (years, days, hours, minutes) to derive hidden insights that inform decision-making. Time series models are very useful models when you have serially correlated data. Most businesses work on time series data to analyze sales projections for the next year, website traffic, competitive positioning and much more.

Learn more about [time series forecasting methods](#), including decompositional models, smoothing-based models, and models including seasonality.

Programming languages used for analyzing time series

Among the many programming languages used for time series analysis and data science are:

- [Python](#)
- [R](#)
- [Java](#)
- SQL
- Julia
- Scala
- MATLAB
- TensorFlow

Identifying time series data

Time series data is unique in that it has a natural time order: the order in which the data was observed matters. The key difference with time series data from regular data is that you're always asking questions about it over time. An often simple way to determine if the dataset you are working with is time series or not, is to see if one of your axes is time.

Time series considerations

Immutability – Since time series data comes in time order, it is almost always recorded in a new entry, and as such, should be immutable and append-only (appended to the existing data). It doesn't usually change but is rather tacked on in the order that events happen. This property distinguishes time series data from relational data which is usually mutable and is stored in relational databases that do online transaction processing, where rows in databases are updated as the transactions are run and more or less randomly; taking an order for an existing customer, for instance, updates the customer table to add items purchased and also updates the inventory table to show that they are no longer available for sale.

The fact that time series data is ordered makes it unique in the data space because it often displays serial dependence. Serial dependence occurs when the value of a datapoint at one time is statistically dependent on another datapoint in another time (read "[Autocorrelation in Time Series Data](#)" for a detailed explanation about this topic).

Though there are no events that exist outside of time, there are events where time isn't relevant. Time series data isn't simply about things that happen in chronological order — it's about events whose value increases when you add time as an axis. Time series data sometimes exists at high levels of granularity, as frequently as microseconds or even nanoseconds. With time series data, change over time is everything.

Different forms of time series data – Time series data is not always numeric — it can be int64, float64, bool, or string.

Time series data vs. cross-sectional and panel data

To determine whether your data is time series data, figure out what you'll need to determine a unique record in the data set.

- If all you need is a timestamp, it's probably time series data.

- If you need something other than a timestamp, it's probably cross-sectional data.
- If you need a timestamp plus something else, like an ID, it's probably panel data.

What the above means becomes clearer upon recalling the definition of (and differences between) each of these three data types:

Time series data definition

Time series data is a collection of **observations** (behavior) for a **single subject** (entity) at **different time intervals** (generally equally spaced as in the case of metrics, or unequally spaced as in the case of events).

For example: Max Temperature, Humidity and Wind (all three behaviors) in New York City (single entity) collected on First day of every year (multiple intervals of time)

The relevance of time as an axis makes time series data distinct from other types of data.

Cross-sectional data definition

Cross-sectional data is a collection of **observations** (behavior) for **multiple subjects** (entities such as different individuals or groups) at a **single point in time**.

For example: Max Temperature, Humidity and Wind (all three behaviors) in New York City, SFO, Boston, Chicago (multiple entities) on 1/1/2015 (single instance)

In cross-sectional studies, there is no natural ordering of the observations (e.g. explaining people's wages by reference to their respective education levels, where the individuals' data could be entered in any order).

For example: the closing price of a group of 50 stocks at a given moment in time, an inventory of a given product in stock at a specific stores, and a list of grades obtained by a class of students on a given exam.

Panel data (longitudinal data) definition

Panel data is usually called as cross-sectional time series data as it is a combination of the above-mentioned types (i.e., **collection of observations for multiple subjects at multiple instances**).

Panel data or longitudinal data is multi-dimensional **data** involving measurements over time. Panel data contains observations of multiple phenomena obtained over multiple time periods for the same firms or individuals. A study that uses panel data is called a longitudinal study or panel study.

For example: Max Temperature, Humidity and Wind (all three behaviors) in New York City, SFO, Boston, Chicago (multiple entities) on the first day of every year (multiple intervals of time).

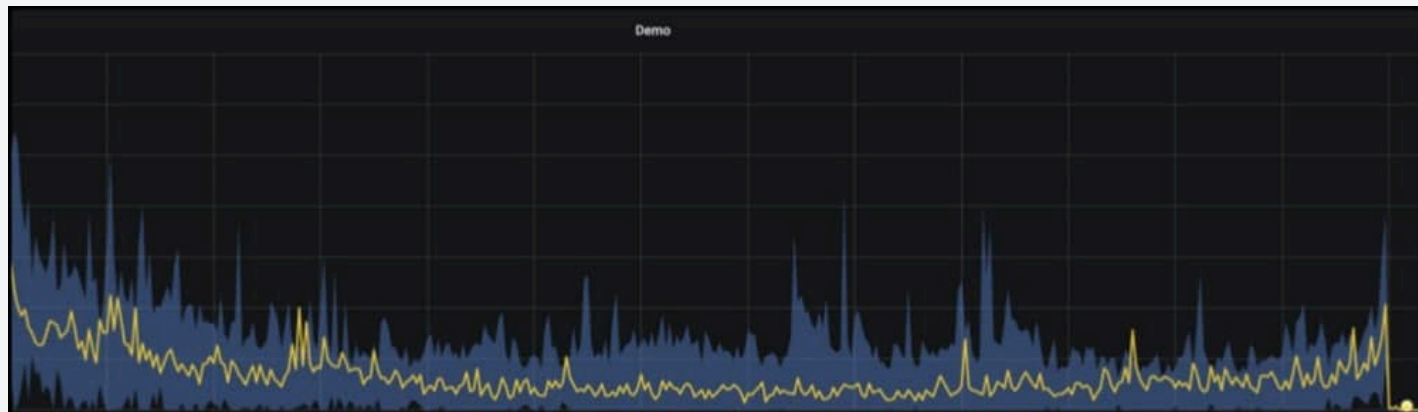
Differences between the three data types

Based on the above definitions and examples, let's recap the differences between the three data types:

1. A **time series** is a group of observations on a single entity over time — e.g. the daily closing prices over one year for a single financial security, or a single patient's heart rate measured every minute over a one-hour procedure.
2. A **cross-section** is a group of observations of multiple entities at a single time — e.g. today's closing prices for each of the S&P 500 companies, or the heart rates of 100 patients at the beginning of the same procedure.
3. If your data is organized in both dimensions — e.g. daily closing prices over one year for 500 companies — then you have **panel** data.

Time series analysis example using InfluxDB

To build a real-time risk monitoring system, Robinhood (a pioneer of commission-free investing) chose **InfluxDB** (a time series database built on open source technologies) and Faust (an open source Python stream processing library). The architecture behind their system involves both time series anomaly detection (InfluxDB) and real-time stream processing (Faust/Kafka).



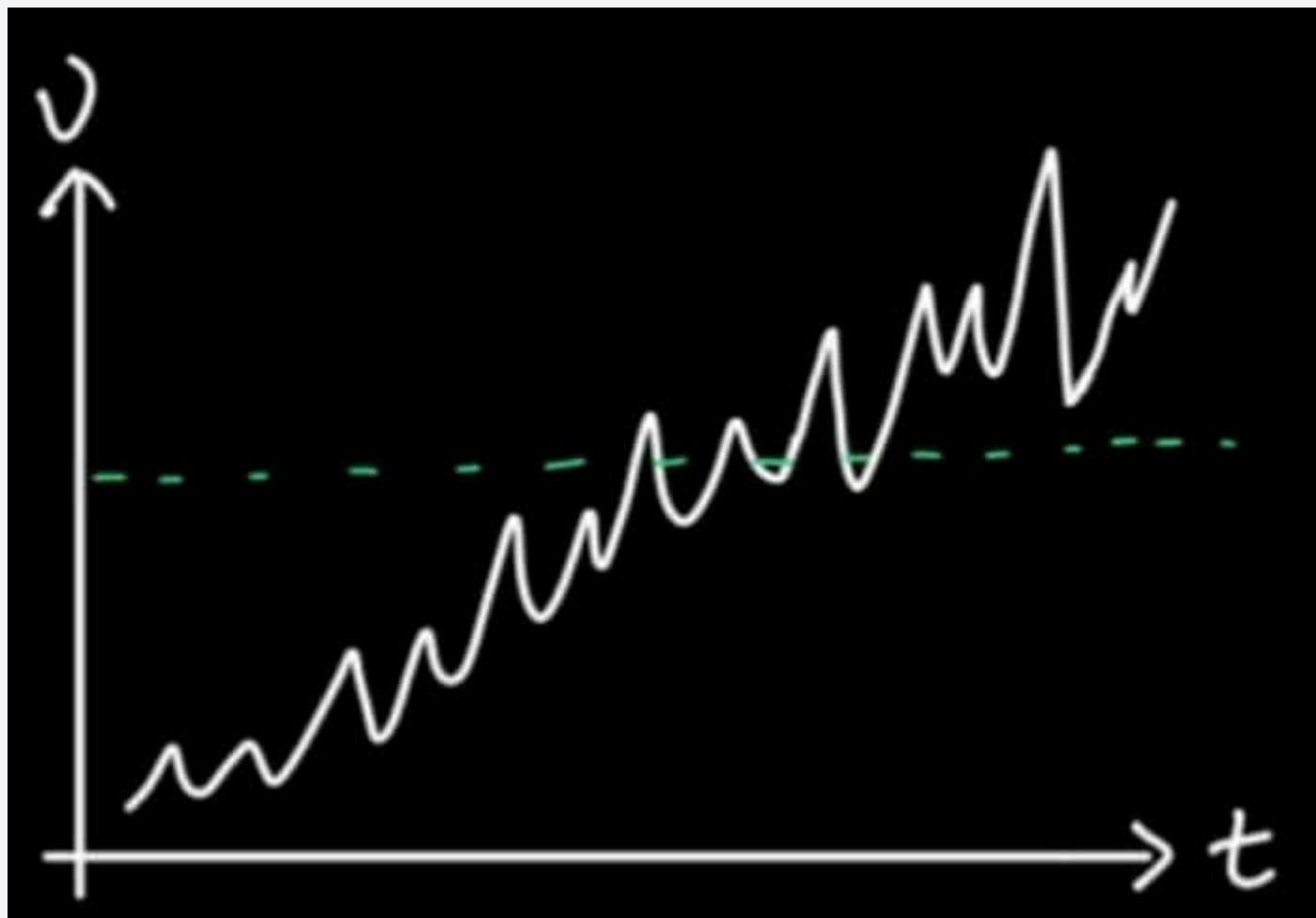
An example of infrastructure telemetry, collected with InfluxDB by Robinhood.

Robinhood alerted on the data with Faust, a real-time Python Library for Kafka Streams.

The aggregated data (yellow) is bounded by upper and lower limits (blue).

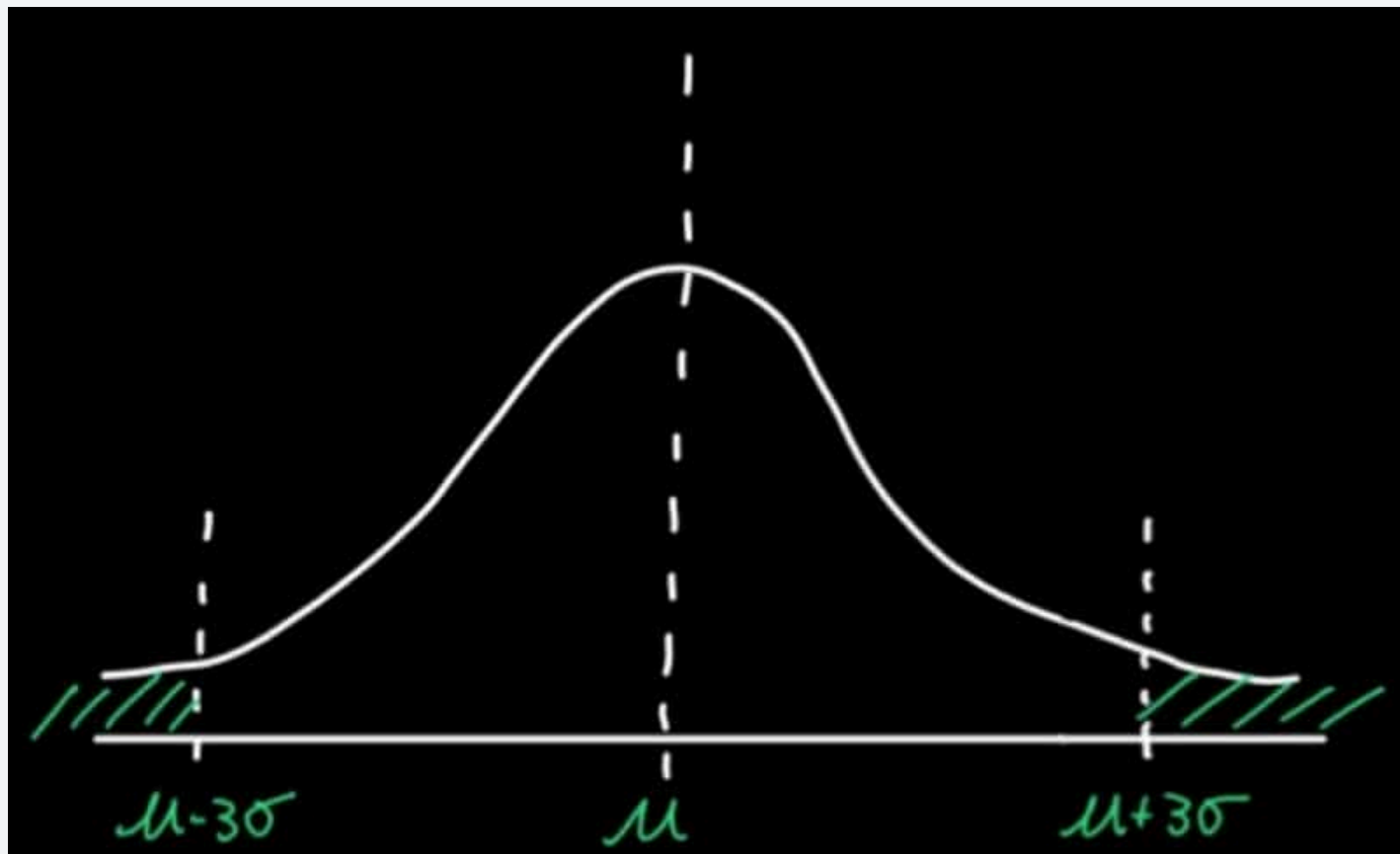
As the number of time series grows, the effort required to understand or detect anomalies in a time series becomes very costly. This is where an anomaly detection system can intelligently alert one when something doesn't go very well.

The first anomaly detection solution that Robinhood tried was threshold-based alerting, by which an alert is triggered whenever the underlying data is over or under the threshold. Threshold-based alerting works well with very simple time series but fails to account for more complex time series. As shown below, the time series here has a trend. It's trending upwards, and there are some up-and-down patterns within that upward trend. If the fixed threshold is used to alert on anomalies, it doesn't work well because it will go over the threshold, and will trigger an alert but will then drop down a threshold and go over a threshold again. So threshold-based alerting in the case of complex time series would require the same effort as checking the dashboard 24/7.



Threshold-based alerting works well with time series but fails to account for seasonality and trend.

To fix this problem, Robinhood alerted on data outside of three standard deviations. Defining your threshold from a standard deviation for anomaly detection is advantageous because it can help you detect anomalies on data that is non-stationary (like the example above). In other words, the threshold defined by a standard deviation will follow your data's trend. Robinhood defined an anomaly as anything outside of three standard deviations away from the mean — so 99.7% of the data lies within this range.



An example of data with a normal distribution. Data that is outside of three standard deviations away from the mean (shaded with green lines) accounts for only 0.03% of all of the data.

Applications in various domains

Time series models are used to:

- Gain an understanding of the underlying forces and structure that produced the observed data.
- Fit a model and proceed to forecasting, monitoring or feedback and feedforward control.

Applications span sectors such as:

- Budgetary analysis
- Census analysis
- Economic forecasting
- Inventory studies
- Process and quality control
- Sales forecasting
- Stock market analysis
- Utility studies
- Workload projections
- Yield projections

Understanding data stationarity

Stationarity is an important concept in time series analysis. Many useful analytical tools and statistical tests and models rely on stationarity to perform forecasting. For many cases involving time series, it's sometimes necessary to determine if the data was generated by a stationary process, resulting in stationary time series data. Conversely, sometimes it's useful to transform a non-stationary process into a stationary process in order to apply specific forecasting functions to it. A common method of stationarizing a time series is through a process called differencing, which can be used to remove any trend in the series which is not of interest.

Stationarity in a time series is defined by a constant mean, variance, and autocorrelation. While there are several ways in which a series can be non-stationary (for instance, an increasing variance over time), a series can only be stationary in one way (when all these properties do not change over time).

Patterns that may be present within time series data

The variation or movement in a series can be understood through the following three components: trend, seasonality, and residuals. The first two components represent systematic types of time series variability. The third represents statistical noise (analogous to the error terms included in various types of statistical models). To visually explore a series, time series are often formally partitioned into each of these three components through a procedure referred to as time series decomposition, in which a time series is decomposed into its constituent components.

Trend

Trend refers to any systematic change in the level of a series — i.e., its long-term direction. Both the direction and slope (rate of change) of a trend may remain constant or change throughout the course of the series.

Seasonality

Unlike the trend component, the seasonal component of a series is a repeating pattern of increase and decrease in the series that occurs consistently throughout its duration. Seasonality is commonly thought of as a cyclical or repeating pattern within a seasonal period of one year with seasonal or monthly seasons. However, seasons aren't confined to that time scale — seasons can exist in the nanosecond range as well.

Residuals

Residuals constitute what's left after you remove the seasonality and trend from the data.

Critical methods of analyzing time series data

Time series analysis methods may be divided into two classes:

- Frequency-domain methods (these include spectral analysis and wavelet analysis) In electronics, control systems engineering, and statistics, the frequency domain refers to the analysis of mathematical functions or signals with respect to frequency, rather than time.
- Time-domain methods (these include autocorrelation and cross-correlation analysis) Time domain refers to the analysis of mathematical functions, physical signals or time series of economic or environmental data, with respect to time. (In the time domain, correlation and analysis can be made in a filter-like manner using scaled correlation, thereby mitigating the need to operate in the frequency domain.)

Additionally, time series analysis methods may be divided into two other types:

- Parametric: The parametric approaches assume that the underlying stationary stochastic process has a certain structure which can be described using a small number of parameters (for example, using an autoregressive or moving average model). In these approaches, the task is to estimate the parameters of the model that describes the stochastic process.
- Non-parametric: By contrast, non-parametric approaches explicitly estimate the covariance or the spectrum of the process without assuming that the process has any particular structure.

Below is an overview of each of the above-mentioned methods.

Spectral analysis

Many time series show periodic behavior that can be very complex. Spectral analysis is a technique that allows us to discover underlying periodicities — it is one of the most widely used methods for data analysis in geophysics, oceanography, atmospheric science, astronomy, engineering, and other fields.

The spectral density can be estimated using an object known as a periodogram, which is the squared correlation between our time series and sine/cosine waves at the different frequencies spanned by the series. To perform spectral analysis, the data must first be [transformed from time domain to frequency domain](#).

Wavelet analysis

What is a Wavelet? A wavelet is a function that is localized in time and frequency, generally with a zero mean. It is also a tool for decomposing a signal by location and frequency. Consider the Fourier transform: A signal is only decomposed into its frequency components.

[Wavelets are analysis tools](#) mainly for time series analysis and image analysis (not covered here). As a subject, wavelets are relatively new (1983 to present) and synthesize many new/old ideas.

Autocorrelation

What is [autocorrelation in time series data](#)? Autocorrelation is a type of serial dependence. Specifically, autocorrelation is when a time series is linearly related to a lagged version of itself. When you have a series of numbers where values can be predicted based on preceding values in the series, the series is said to exhibit autocorrelation. By contrast, correlation is simply when two independent variables are linearly related.

Here's why autocorrelation matters. Often, one of the first steps in any data analysis is performing regression analysis. However, one of the assumptions of regression analysis is that the data has no autocorrelation. This can be frustrating because if you try to do a regression analysis on data with autocorrelation, then your analysis will be misleading.

Additionally, some time series forecasting methods (specifically regression modeling) rely on the assumption that there isn't any autocorrelation in the residuals (the difference between the fitted model and the data). People often use the residuals to assess whether their model is a good fit while ignoring that assumption that the residuals have no autocorrelation (or that the errors are independent and identically distributed or i.i.d). This mistake can mislead people into believing that their model is a good fit when in fact it isn't.

Finally, perhaps the most compelling aspect of autocorrelation analysis is how it can help us uncover hidden patterns in our data and help us select the correct forecasting methods. Specifically, we can use it to help identify seasonality and trend in time series data. Additionally, analyzing the autocorrelation function (ACF) and partial autocorrelation function (PACF) in conjunction is necessary for selecting the appropriate ARIMA model for your time series prediction. Learn [how to determine if your time series data has autocorrelation](#).

Cross-correlation

Cross correlation is a measurement that tracks the movements of two variables or sets of data relative to each other. In its simplest version, it can be described in terms of an independent variable, X, and two dependent variables, Y and Z. If independent variable X influences variable Y and the two are positively correlated, then as the value of X rises so will the value of Y.

If the same is true of the relationship between X and Z, then as the value of X rises, so will the value of Z. Variables Y and Z can be said to be cross correlated because their behavior is positively correlated as a result of each of their individual relationships to variable X.

Parametric vs. nonparametric tests

Parametric tests assume underlying statistical distributions in the data. Therefore, several conditions of validity must be met so that the result of a parametric test is reliable. Nonparametric tests are more robust than parametric tests. They are valid in a broader range of situations (fewer conditions of validity).

Nonparametric tests do not rely on any distribution. They can thus be applied even if parametric conditions of validity are not met. Parametric tests will have more statistical power than nonparametric tests. A parametric test is more able to lead to a rejection of H_0 . Most of the time, the p-value associated to a parametric test will be lower than the p-value associated to a nonparametric equivalent that is run on the same data.

Time series analysis best practices

For the best results in terms of time analysis, it's important to gain a better understanding of exactly what you're trying to do in the first place. Remember that in a time series, the independent variable is often time itself and you're typically using it to try to predict what the future might hold.

To get to that point, you have to understand whether or not time is stationary, if there is seasonality, and if the variable is autocorrelated.

Autocorrelation is defined as the similarity of observations as a function of the amount of time that passes between them. Seasonality takes a look at specific, periodic fluctuations. If a time series is stationary, its own statistical properties do not change over time. To put it another way, the time series has a constant mean and variance regardless of what is happening with the independent variable of time itself. These are all questions that you should be answering prior to the performance of time series analysis.

Time series models

Generally speaking, there are three core models that you will be working with when performing time series analysis: autoregressive models, integrated models and moving average models.

An autoregressive model is one that is used to represent a type of random process. It is most commonly used to perform time series analysis in the context of economics, nature and more.

Moving-average models are commonly used to model univariate time series, as the way the output variable is presented depends linearly on both the current and past values of an imperfectly predictable term. A traditional integrated model is one that lists all data points in time order.

How to do time series analysis will obviously vary depending on the model you choose to work with.

Learn how Nobl9 Saved time and money with InfluxDB

01:48

Frequently asked questions (FAQ) about time series data

Where is time series data stored?

Time series data is often ingested in massive volumes and requires a [purpose-built database](#) designed to handle its scale. Properties that make time series data very different than other data workloads are data lifecycle management, summarization, and large range scans of many records. This is why time series data is best stored in a [time series database](#) built specifically for handling metrics and events or measurements that are time-stamped.

Learn more about time series data storage and about the [best way to store, collect and analyze time series data](#).

01:23

What is a time series statistic?

A time series statistic refers to the data extracted from a [time series model](#). The information must be recorded over regular time intervals, and may be combined with cross-sectional data to derive relevant predictions.

What are time plot statistics?

Time plot statistics refer to the evolution of a series over a specific time interval. It's often used at the beginning of an analysis for quick interpretation of anything from trends to anomalies.

Is a time series database the same as a data warehouse or data lakehouse?

No, but the [data warehousing](#) process helps you analyze data that exists in all the systems and software tools in your organization, including a time series database. Read more on [how InfluxDB works with your data lakehouse](#).

How is time series data understood and used?

Time series data is gathered, stored, visualized and analyzed for **various purposes across various domains**:

1. In data mining, pattern recognition and machine learning, time series analysis is used for clustering, classification, query by content, anomaly detection and forecasting.
2. In signal processing, control engineering and communication engineering, time series data is used for signal detection and estimation.
3. In statistics, econometrics, quantitative finance, seismology, meteorology, and geophysics the time series analysis is used for forecasting.

#1 Time Series Database

According to DB Engines

Join the millions of developers using InfluxDB to predict, respond, and adapt in real-time.

[Get InfluxDB](#)



Developer Education

Training for time series app developers.

[View All Education](#)

Learn more about InfluxDB

Performance Benchmarking: InfluxDB 3.0 vs. InfluxDB Open Source

[Explore Benchmarks](#) →

InfluxDB for Industrial IoT: A Live Demonstration

[Watch Demo](#) →

How Time Series Databases and Data Lakes Work Together

[Read Article](#) →

Time Series Forecasting: 2025 Methods and Complete Guide

[Read Guide](#) →

Network Monitoring

[See Solutions](#) →

Time Series Data Analysis: Definitions and Best Techniques in 2024

[Read Tech Paper](#) →



Product & Solutions

InfluxDB
InfluxDB Cloud
Serverless
InfluxDB Cloud
Dedicated
InfluxDB Clustered
InfluxDB Comparison
Integrations
Data Lake /
Warehouse
Data Collector
Pricing
Use Cases
Time Series Data

Developers

Guides
Blog
Customers
Support
Webinars
Documentation
Events & Live
Training
InfluxDB University
Community
InfluxDB OSS
Telegraf
AWS
Product Integrations
Glossary

Company

About
Careers
Partners
Newsroom
Contact Us
Customers

Sign up for the InfluxData newsletter

[Time Series](#)[Database](#)[Time Series](#)[Forecasting](#)[Data Warehousing](#)[Network Monitoring](#)

548 Market St, PMB 77953

San Francisco, California 94104

Follow Us



© 2025 InfluxData Inc. All Rights Reserved.

[Legal](#) [Security](#) [Cookie Policy](#) [Comparison](#)