|  |  |
| --- | --- |
| Methods to calculate Confidence intervals  With a numeric-oriented approach |  |

Daniel Abutbul

This report was written as a self-learning exercise

Contents

[Motivation 2](#_Toc106569568)

[Working example 2](#_Toc106569569)

[Extrapolation and Confidence Intervals 3](#_Toc106569570)

[Other literature 4](#_Toc106569571)

[Exact solution 6](#_Toc106569572)

[∆*χ*2 approach of acceptable models 9](#_Toc106569573)

[*χ*2 reasoning and systematic noise 9](#_Toc106569574)

[∆*χ*2 distribution 9](#_Toc106569575)

[Monte Carlo Simulation of prediction’s pdf 10](#_Toc106569576)

[Minimal Δ*χ*2 best fit 10](#_Toc106569577)

[With Robust Fit 10](#_Toc106569578)

[Bayesian 11](#_Toc106569579)

[Sources 12](#_Toc106569580)

# Motivation

For many applications, we encounter the problem of creating predictions based on observed data. While a very general case can be formulated, if one wishes to practice, understand, implement and finally use such predictions, it is best first to solve the simplest imaginable problem, and try and solve it in as many methods as possible, while pointing out their advantages and disadvantages.

In this report, we will review and implement various methods for extrapolating the value of a straight line calibrated from observed data.

We will assume certain knowledge in probability theory and linear regression.

## Working Example

Throughout this report, we will use the same simple example of the straight line:

The straight line formula tells us exactly the value of at any value of . For many real life situation, though, the formula and its parameters are unknown to the researcher. Instead, the researcher may have a sample of points which lie near the straight line, and originated from the straight line, but with added noise. This can represent for example a scientific experiment of measuring the value of at different ’s, with added measurement noise.

The sample is represented by a set of points , at each one we have the value . For our simplest working example, we assume the noise comes from a normal distribution, and we model it using a random variable , such that is sampled normally around the true value :

It is very common, and still useful, to restrict ourselves to the case that are i.i.d. variables:

Finally, our very specific working example will be of the data set:

The straight line, a possible sample points with noise from the straight line, and theirs best fit ( minimized) are given in Figure 1:

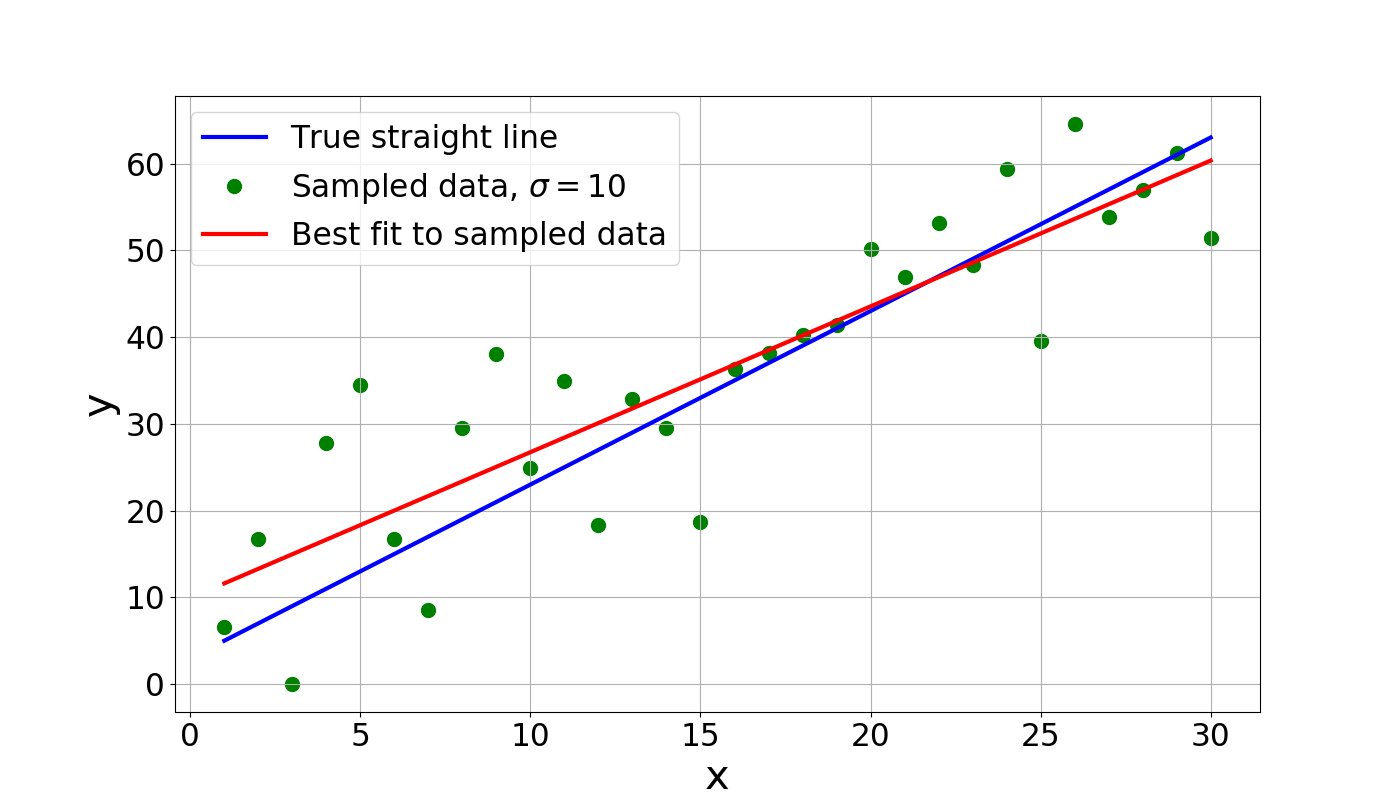


Figure 1: The straight line , sampled with normal noise ( i.i.d. random variables) and fitted with a straight line.

## Extrapolation and Confidence Intervals

Our task is to construct an extrapolation toward a value that is not in our data. In this report we will try to extrapolate the value of the straight line toward:

We will not only predict the value of at (that would most likely simply be the best fit to the observed data). We will decide on some range around the extrapolated value, such that in most noise realization the true value of the straight line will be within our predicted range, as demonstrated in Figure 2:

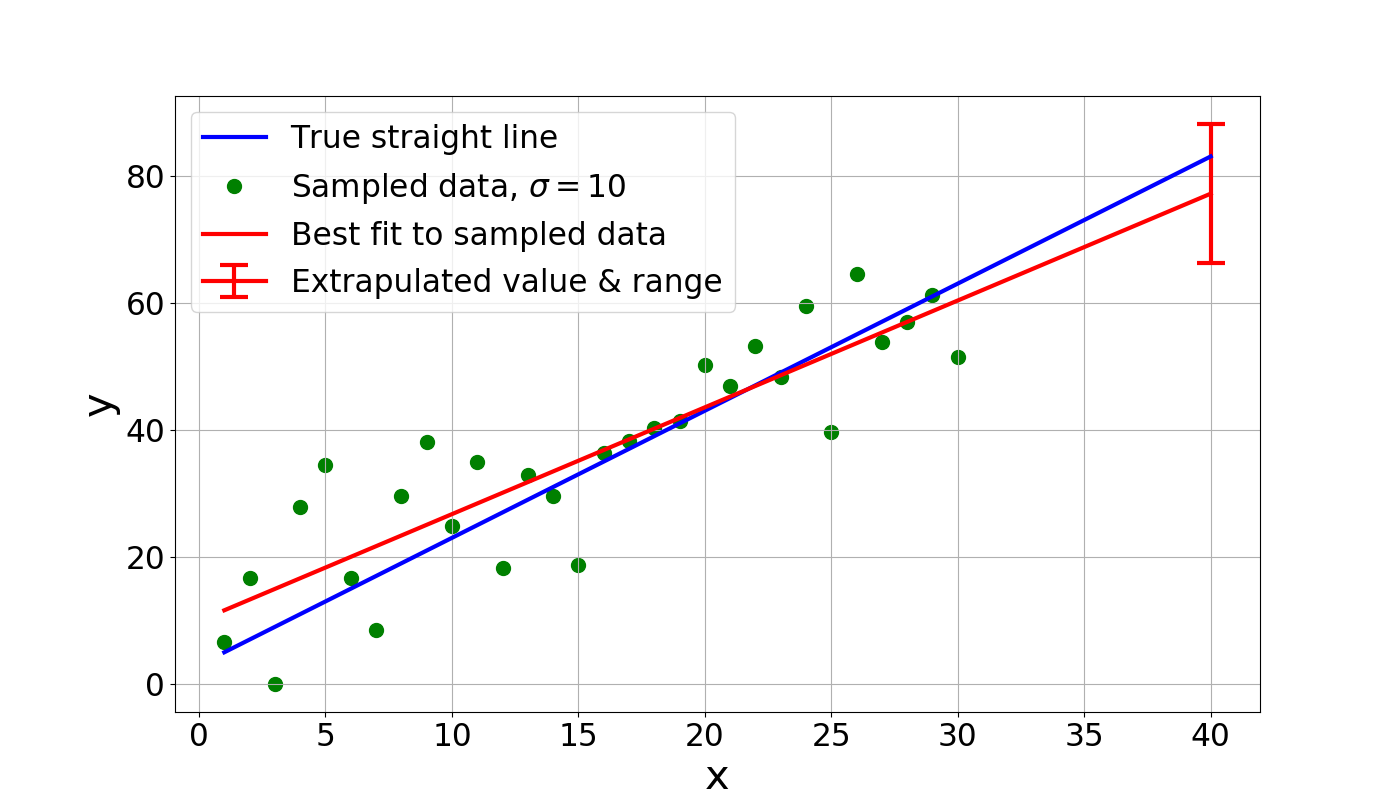


Figure 2: Extrapolation from sample x=1…30 toward and some range around it. We choose the range such that for most noise realizations, the true value of the straight line will fall within our range.

The Confidence Interval (CI) is a range around that extrapolated point, **constructed based on the sampled data**. We will say a range is a 95%CI if the true value falls within the range with confidence level (CL) of 0.95. Notice a 95%CI changes every noise realization; for each realization, we construct a range based on the observed sampled data. For each realization, the true value will either be inside or outside that range.

When we know the true line, which rarely happens in real life but does happen in this report, we can test our method for constructing 95%CI by running it for different noise realization. For each 95%CI construction method, we test if indeed in 95% of noise realizations the true line falls within that range. That is, if indeed the CL is 95%.

In general, though, since we have only the sampled data and we construct some range around it, we can never know for sure if the true line indeed falls within the constructed range. When we call a range a 95%CI in such a case, we actually **model** the noise, and then construct a range such that we believe the CL is 95%, that is the probability the true line is inside that range is 0.95.

## Other Literature

There is a lot of literature on the subject. The goal of this report is also to implement and practice the different methods, and so it does not claim novelty over existing literature (and definitely does not put as a goal to resemble a scientific peer-reviewed paper). Some main sources used in the preparation of this report are ‎[1]-[TODO].

# Exact Solution

Conveniently, the problem of calculating the 95%CI for the case of extrapolating a straight line has an exact solution. Since our goal is to test methods for CI construction in the simplest environment we can, having an exact solution is a great benefit.

Why then should we follow up in next chapters with all the other methods? Because the exact solution we will construct works only for the extrapolation of the straight line. For example, if we want to predict the value of some target function, say ( are the straight line parameters) the exact solution does not apply.

To construct the exact solution, we will use the linear regression language, for which a general model that is linear in its parameters is given by:



( is an vector and is vector. The result is a scalar)

represent the sample point at which we calculate (notice for a straight line: ), and are the parameters of the linear model. For the straight line we have and so . As mentioned above, we wish to investigate the case when we sample from a linear model with added noise, modeled by the normal i.i.d. random variables . In vector notion, we rewrite equation ‎(2) in the form:

is called the design matrix. For samples of a straight line at points we have . The best fit as function of the noise is given by ‎[2]:

( is an matrix, and is an square matrix which has inverse. is matrix, and is matrix, and so the result is an matrix)

Our best fit prediction to any point is given using equation ‎(6) with the replacement of the true parameters by the fitted ones :

Since our goal is to determine a range around our prediction, it is natural to calculate the variance of . Also notice since are linear combination of the normal random variables , they are themselves normally distributed.

To calculate the variance, another useful result from linear regression models is that ‎[3], and so:

Since one can see that , and so to summarize is a normal random variable with mean and variance given by:



In normal distributions a range of around the mean covers 95.4% of the area under the pdf, and so it will be most convenient to calculate the 95.4%CI throughout the report. Formula ‎(10) gives us then the 95.4%CI around any prediction constructed symmetrically around , since the probability for to fall within a range of from the true is 0.954. We can plot the 95.4%CI range as function of (for the same sample data from Figure 2), as shown in Figure 3:

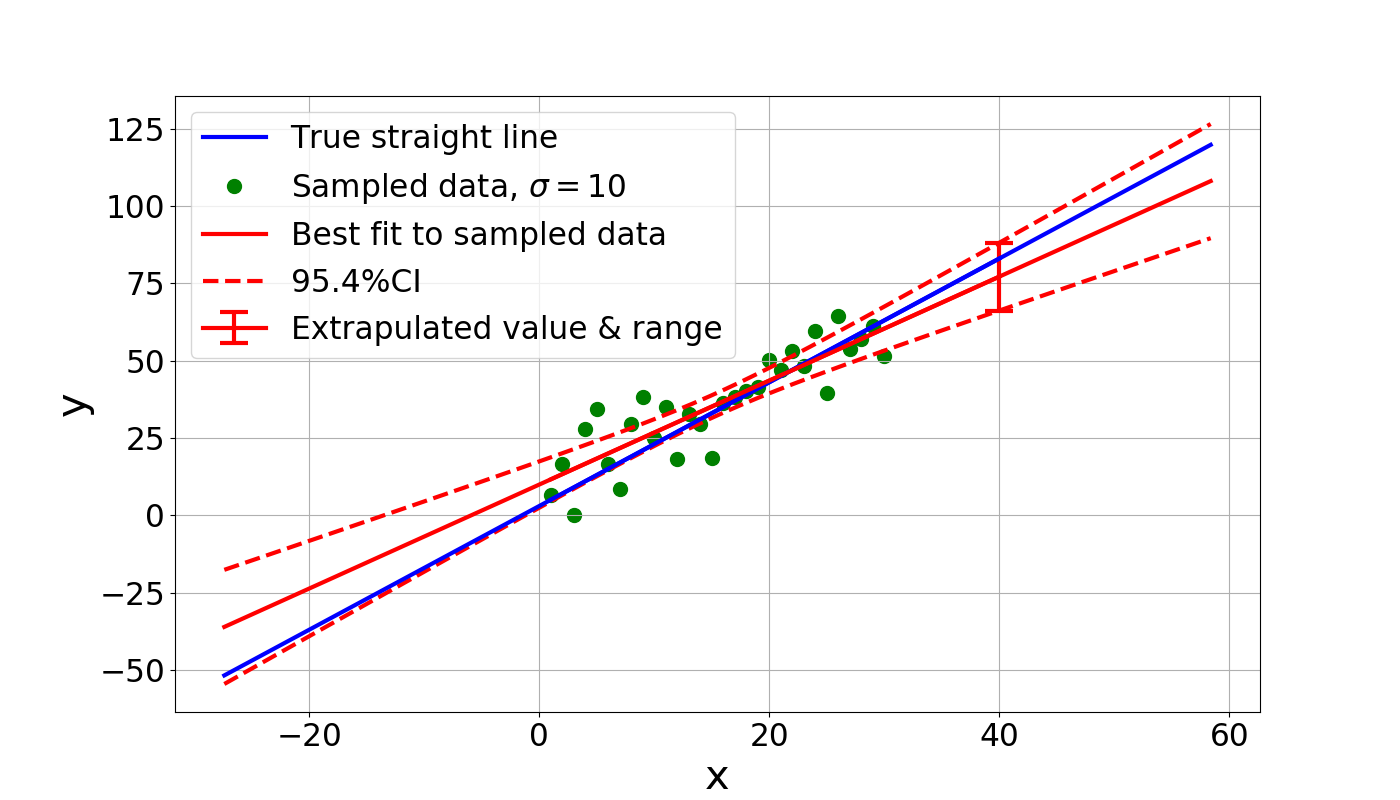


Figure 3: CI (95.4%CI) around the best fit. It can be seen that the prediction is more accurate around the middle of the sample, but as the extrapolation deviate from the sample the uncertainty rises.

The error bar at is of size 21.9, that is . The standard deviation is about half the size of our original noise’s standard deviation .

For any CI calculation method for which the range magnitude is independent of the noise realization, like in the case of the exact solution, we should then get the same CI that was calculated exactly here, that is a range of size 21.9.

We can still run our test and see that indeed the true value of falls within the constructed range in 95.4% of cases. Using 100,000 monte carlo realizations, in 95.3% the true value falls within the CI range, which means that: a. our exact solution is correct and b. from 100,000 Monte Carlo realizations we can expect an error of about 0.1% in the CL[[1]](#footnote-1).

# ∆*χ*2 approach of acceptable models

## *χ*2 reasoning and systematic noise

## ∆*χ*2 distribution

# Monte Carlo Simulation of Prediction’s PDF

## Minimal Δ*χ*2 Best Fit

TODO: It is completely pointless seen we have proven above that the CI range is independent of the nosie realization

## With Robust Fitting

TODO: running MC here will be heavy, as for each noise realization one need to perform other MC realizations to calculate the CI range. Maybe the CI range is independent of the noise realization? It is possible… In such case calculating the CI range once will be quite enough?

TODO: Choose robust fitting – median? Probably there is something very common in the literature to use here…

# Boot Strapping with Studentized Residual Resampling

TODO: See ‎[6]. Use Studentized as they are equally distributed, while the residuals are not (have different variance…)

# Bayesian

TODO: use <https://www.pymc.io/projects/docs/en/latest/installation.html>

# Sources

1. “Numerical Recipes, the Art of Scientific Computing”, Third Edition, <https://e-maxx.ru/bookz/files/numerical_recipes.pdf> . William H. Press, Saul A. Teukolsky, William T. Vetterling & Brian P. Flannery, Cambridge university press
2. Wikipedia linear regression, <https://en.wikipedia.org/wiki/Linear_regression>
3. “Statistics 512: Applied Linear Models; Chapter 5: Linear Regression in Matrix Form”, <https://www.stat.purdue.edu/~boli/stat512/lectures/topic3.pdf> , Bo Li, associate Professor, [Department of Statistics](http://www.stat.purdue.edu/), [Purdue University](http://www.purdue.edu/)
4. Benjamin T. Vincent . "Bayesian regression with truncated or censored data". In: *PyMC Examples*. Ed. by PyMC Team. DOI: [10.5281/zenodo.5654871](https://doi.org/10.5281/zenodo.5654871), <https://www.pymc.io/projects/examples/en/latest/generalized_linear_models/GLM-truncated-censored-regression.html>
5. <https://en.wikipedia.org/wiki/Confidence_interval>
6. <https://en.wikipedia.org/wiki/Bootstrapping_(statistics)>

1. In the python implementation, a 95.4% CL was given exactly as an input, and the number of standard deviation used to construct the CI was calculated to high precision, and so the number of standard deviation was corrected from 2.0 to 1.9954 [↑](#footnote-ref-1)