Mathematical, Numerical, and Statistical Methods in Extraterrestrial Physics

. . . currently with a focus on statistial methods

Lecture Notes
Winter 2017/2018
Summer 2022

Robert F. Wimmer-Schweingruber
Christian-Albrechts-Universität zu Kiel
Institut für Experimentelle und Angewandte Physik
Abt. Extraterrestrik
Leibnizstrasse 11, D-24118 Kiel

April 18, 2023

Contents

Chapter 1

Introduction

Physicists are sometimes described as machines that turn coffee into mathematical formulae. Unfortunately, coffee is not enough, and courses in the normal physics curriculum normally do not prepare students well enough for some of the challenges they face when analyzing problems in (extraterrrestrial) physics. For instance, students learn rudimentary concepts of statistics in elementary lab courses, especially in view of the correct treatment of errors. They often do not, however, learn how to derive confidence limits of model predictions or parameters. In the past 10, 15 years, I have found that there is an urgent need for a course in more advanced methods in the statistical analysis of data. Similarly, I have found that other important methods for data analysis are sorely missing in physics curricula, as are many numerical methods which ought to be taught to modern, computer-literate students. Such considerations inspired this course, and I hope that it will be useful to the reader.

Some of the inspiration for these lecture notes also came from the free EBook by *Brink* (2010) which gives a consise summary of statistical methods but also adds a number of helpful examples which other text are too often too skimpy with. I also profited from the highly readable and practical-minded explanations given in *Press et al.* (1989), and have made heavy use of Wikipedia and several online resources listed below. I gladly acknowledge countless discussions with many students and Post-Docs, colleagues, and my teachers.

List of online resources:

`http://www.itl.nist.gov/div898/handbook/index.htm`
`https://onlinecourses.science.psu.edu/stat414`
`https://onlinecourses.science.psu.edu/stat415`
`http://numerical.recipes/`

Needless to say that I used the SAO/NASA Astrophysics Data System (ADS) extensively for these lecture notes – should you not yet know it, here's the link:

`https://ui.adsabs.harvard.edu/`

The lecture notes you are looking at are a growing experiment and will hopefully become more complete and useful with time. They begin with a short summary of some important concepts used in space physics and continues with some useful numerical methods. The main part of this course focused on the statistical analysis of data for which the previous two chapters serve to provide the physical models. It begins with a summary of the most important concepts in probability. Some of these concepts are applied to the field of random variables and some of the most important distributions. It is humbling to a physicist that the outcome of his most carefully designed experiment is called a random variable by a statistician! On the other hand, it is gratifying to then notice that the aim of the experiment is to ensure that the distribution function which describes the experimental outcome, i.e., the measurements, is strongly peaked and minimizes the number and range of the relevant input parameters. Methods how to generate random numbers are discussed and some examples are shown in chapter 6, and methods to describe distributions (descriptive statistics) are given in the following chapter 7.

TODO Motre introductory text on all material covered. Add hypothesis testing and fitting.

Kiel, summer 2022

Chapter 2

The Measurement Process

Figure 2.1 shows properties (such as the half life) of particles (such as the neutron) vs. their publication date in the Review of Particle Properties (*Patrignani et al.*, 2016). One immediately sees two striking features: The values change with time and the error bars decrease with time. A little more thought shows a third striking feature – the values with the large error bars are often quite different from the ones with small error bars, loosely spoken, they are inconsistent with later values.

Figure 2.1 also shows an important lesson: Measurement is a tricky business, it is difficult, it is an art. The people who published the values shown in Fig. 2.1 weren't stupid, they published the best results they could at their time. But there is even more to this figure which is not apparent. Take for example the half life of the neutron (upper left sub-figure). *Patrignani et al.* (2016) give the mean life of the free neutron as $\tau = 880.2 \pm 1.0$ seconds. They then write "*We average the best seven measurements. The result,* 880.2 ± 1.0 *s (including a scale factor of 1.9), is 5.5 seconds lower than the value we gave in 2010—a drop of 6.9 old and 5.5 new standard deviations.*" We can now also look at the data which they used to compute the average, it is given in Tab. 2.1. The averaging process is illustrated in Fig. 2.2.

Looking carefully at Tab. 2.1 you will see that the UCN methods seem to give lower values than the two other methods. Indeed, the two newest measurements show $\tau = 880.2 \pm 1.2$ and $\tau = 887.7 \pm 1.2 \pm 1.9$ which appear to be mutually exclusive! This is indeed a puzzle which is not understood. It also shows how important it is to measure well and do your best to report accurate measurement errors or uncertainties. Is there some new physics hidden in this puzzle?

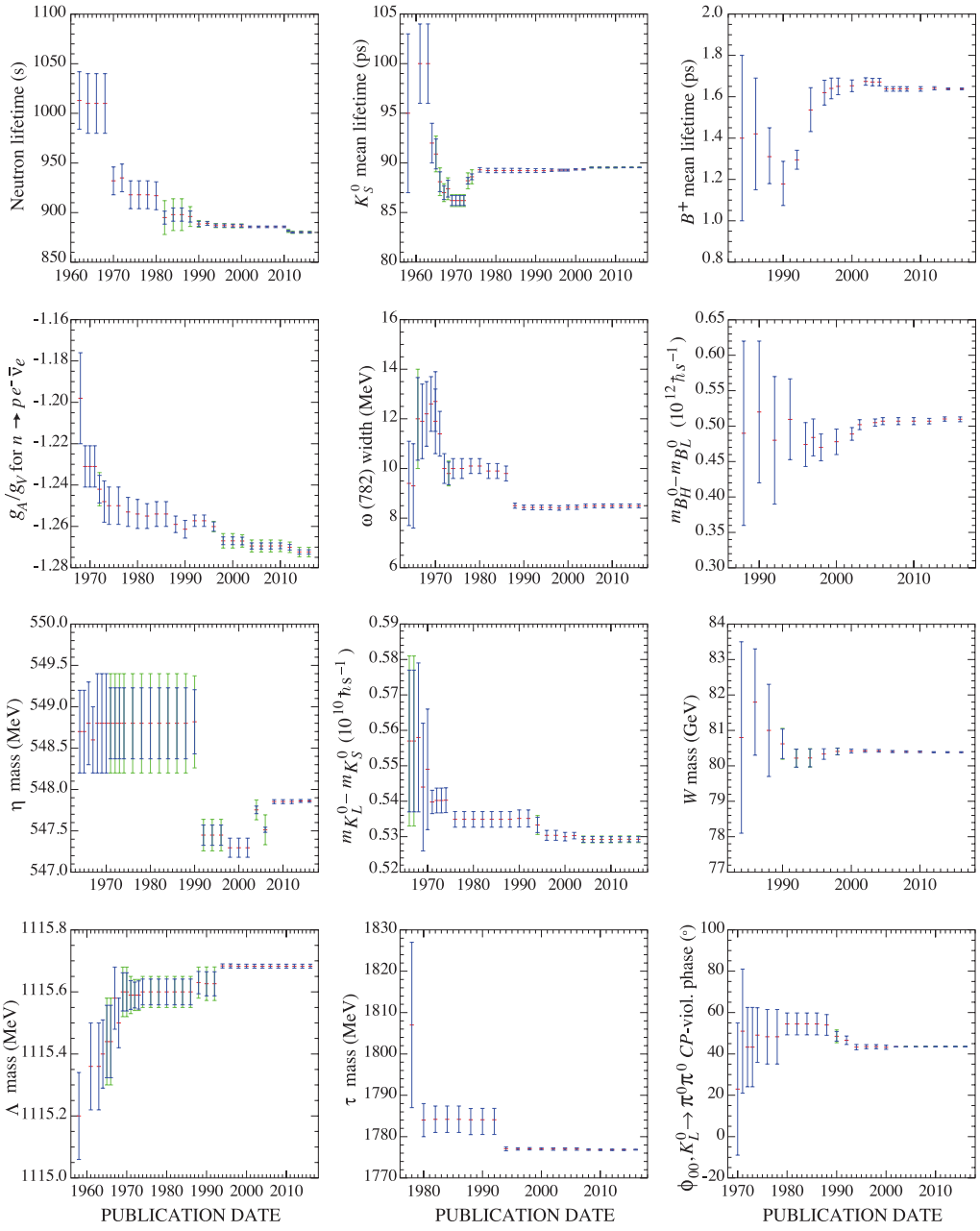


Figure 2: A historical perspective of values of a few particle properties tabulated in this *Review* as a function of date of publication of the *Review*. A full error bar indicates the quoted error; a thick-lined portion indicates the same but without the "scale factor."

Figure 2.1: See caption above. From (*Patrignani et al.*, 2016).

mean life [s]	Source	Method
880.2 ± 1.2	ARZUMANOV 15	UCN double bottle
$887.7 \pm 1.2 \pm 1.9$	YUE 13	In-beam n, trapped p
$882.5 \pm 1.4 \pm 1.5$	STEYERL 12	UCN material bottle
$880.7 \pm 1.3 \pm 1.2$	PICHLMAIER 10	UCN material bottle
$878.5 \pm 0.7 \pm 0.3$	SEREBROV 05	UCN gravitational trap
$889.2 \pm 3.0 \pm 3.8$	BYRNE 96	Penning trap
882.6 ± 2.7	MAMPE 93	UCN material bottle
$\mathbf{880.2 \pm 1.0}$	Error includes scale factor of 1.9.	

Table 2.1: Neutron mean life times used for the *Patrignani et al.* (2016) average value of 880.2 ± 1.0 seconds.

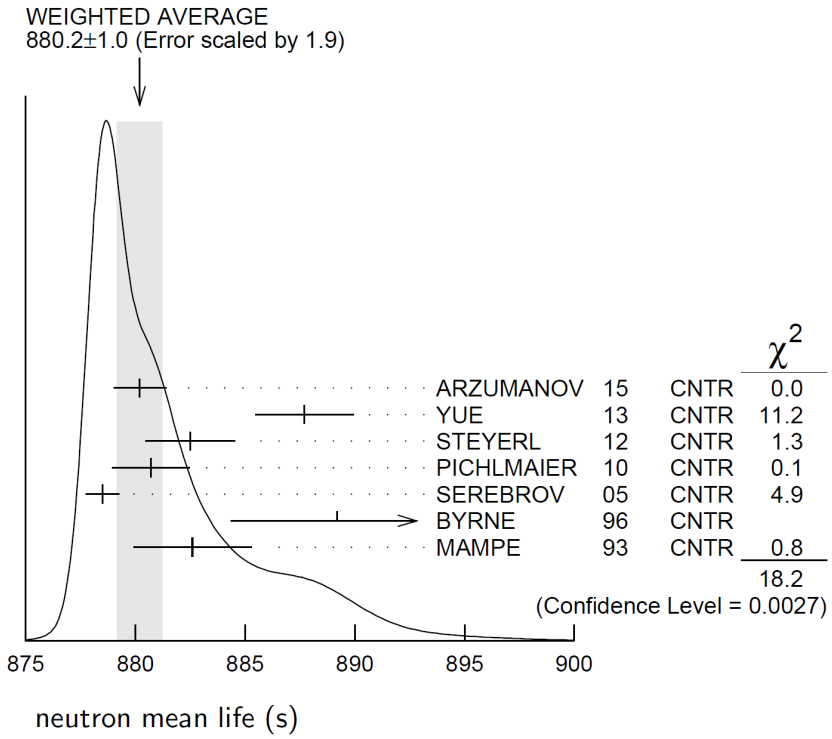


Figure 2.2: Determination of the neutron mean life time. From (*Patrignani et al.*, 2016).

2.1 Measurement Errors/Uncertainties

Wikipedia gives the following definition of measurement error: "*Observational error (or measurement error) is the difference between a measured value of a quantity and its true value. In statistics, an error is not a "mistake." Variability is an inherent part of the results of measurements and of the measurement process.*"

There is an important point to keep in mind about the concept of measurement and that is that we do not know the "true value"!

We generally distinguish between two kinds of error, *statistical* and *systematic* errors. Let us first consider systematic errors. These can be due to an uncertainty in the calibration of the measurement equipment. They are reproducible, i.e., they will always be the same and give a measured value which is always too large or too small. This kind of error needs to be estimated based on your knowledge of your equipment.

Example 2.1. If we measure the width and height of a door we use a folding ruler which has little lines on it with a distance of 1 mm. We assume that this means that it has a length of 2 ± 0.001 meters. But how long has it been in use and in what conditions? Did it get wet and expand? Or did it get too dry and contract? To be conservative we might give a systematic error of ± 2 mm.

Note that *systematic* uncertainties are not the same as a *bias*! If you know that your equipment systematically measures less than the "true value", you need to correct for this systematic bias. If you know your stop watch runs 1% too fast, you can easily correct the measured times and reduce them to the "correct" measured time.

Statistical uncertainties or errors are not reproducible. If you repeat your experiment many times, you will find that the results differ slightly. This can be due to unknown sources of errors (e.g., you did not take into account temperature changes) or fundamental limitations of your equipment, e.g., electronic noise in your detector or Heisenberg's uncertainty principle. If the latter is your largest source of errors - Congratulations!

2.2 Accuracy vs. Precision

The **accuracy** of an experiment is a measure of how close your result is to the "true value". The **precision** of an experiment is a measure of how exactly it measures with no consideration of possible bias. Figure 2.3 summarizes the meanings of accuracy and precision. Obviously you should always aim at being as precise and accurate as possible, only one of the two is not sufficient. The accuracy of the

experiment is determined by its systematic errors, the precision by its statistical errors.

The precision of a result is sometimes implied in the reported numbers rather than reported as an error or uncertainty. On can indicate the precision by the number of significant digits. This is defined as follows (*Bevington* and *Robinson*, 2003):

1. The leftmost non-zero digit is the most significant digit.
2. If there is no decimal point, the right-most non-zero digit is the least significant digit.
3. If there is a decimal point, the right-most digit is the least significant digit, even if it is zero.
4. All digits between the least and most significant digits are counted as significant digits.

Example 2.2. The following numbers all have four significant digits: 1234; 123400; 123.4; 1001; 10.10; 0.0001010; 100.0

When you report a result, the number of significant digits should typically be one more than allowed by the experimental precision. You should round your result to the thus defined number of significant digits using the usual rounding rules.

2.3 Understanding Your Measurement Equipment

In order to provide estimates of the systematic and statistical errors of our experiment, we need to understand the measurement equipment. This can be done in various ways.

2.3.1 Calibration

Using a known calibration standard, you can repeat the same measurement multiple times. This gives you information about the calibration of your equipment, for example how it maps the "true" calibration standard to your measurements,

$$m : \mathbb{R} \longrightarrow \mathbb{R}.$$

In an ideal world, the function m (for measurement) will be linear. Its slope may not be unity and its intercept may also not necessarily be zero, but you can determine these values from calibration. In addition to these two quantities, you

can also determine the statistical errors by repeating the measurement with the same calibration standard multiple times. The more often you measure the same standard, the better you will know your statistical errors. Of course this can get rather expensive, so don't be too greedy.

With enough statistics (and many, many hours spent in calibration facilities) you will be able to fit a (mathematical) model of your instrument to the calibration data. Yes, you will not get around having a model of your instrument. You will always need one to understand your calibration data. The simplest possible model for your experiment is a straight line,

$$m(x) = a \cdot x + b,$$

which translates the value of the calibration standard into your measurement value. An example for such a "calibration line" is given in Fig. 2.4. The measurements (y) are plotted together with their errors against the calibration "standards" (x) and we have only allowed for measurement errors but none in the calibration "standards". The grey-shaded area shows the band in which 95% of all future measurements should lie. Note that one data point lies outside this band. That is OK as there are 25 data points and 1 of them is then 4%. The blue shaded area shows the 95% confidence band for the expectation value for future measurements and the red line shows the best fit model. You see that the calibration is best in the middle of the data and is less well defined towards the edges of the data. I will show you how to determine such a calibration line in chapter 9.

2.3.2 Modelling

The discussion leading up to here has hopefully convinced you that to make good measurements you need a good understanding of your equipment and a good model of it too. There are several points which you should consider when modelling your equipment or instrument.

- KISS! (Keep It Simple, Stupid!) Keep your model as simple as possible but include all relevant parameters. Yes, this is called the KISS principle.
- Think ahead! As you work on your model, you are likely to discover more and more effects which you need to account for. Keep your model flexible enough that it can cope with later additions.
- Document your work! Once you've understood your equipment, begin writing a "calibration report". Start with the overall structure, add plots and tables, then descriptions of them, then explanatory text, and you're already nearly done!

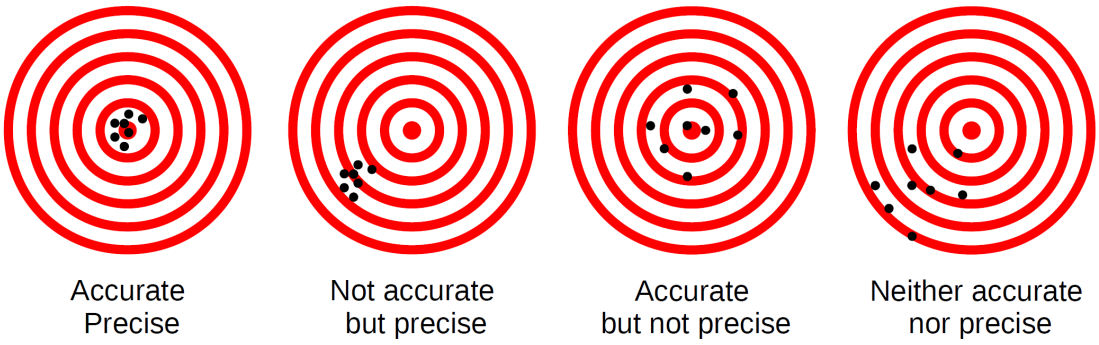


Figure 2.3: Visualization of the meaning of accuracy and precision.

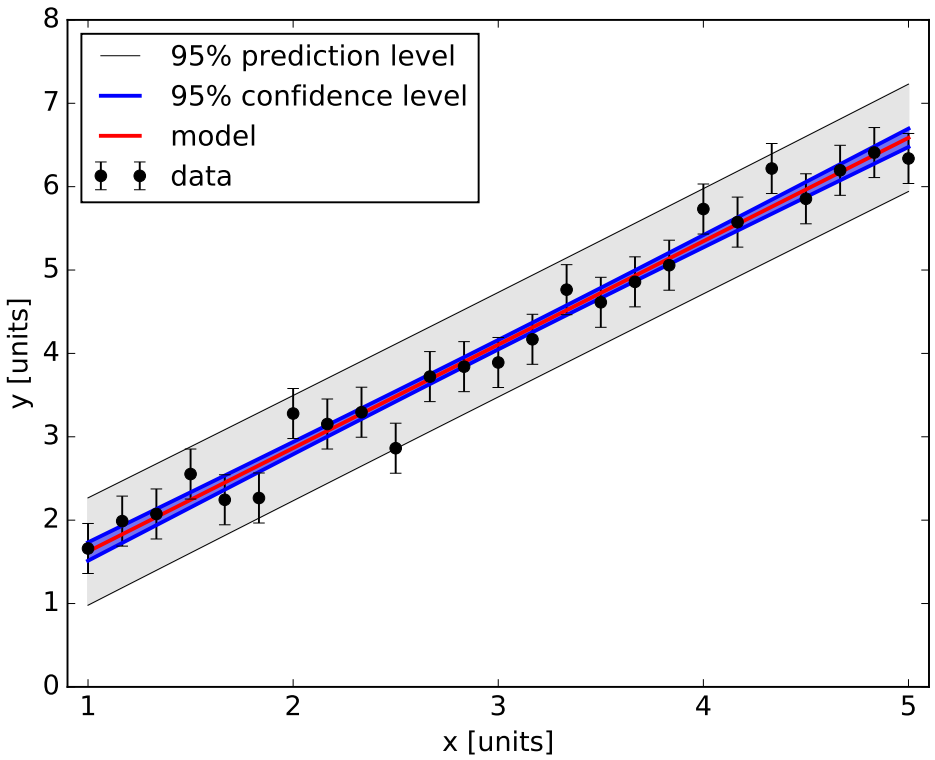


Figure 2.4: Example of a calibration line. Calibration "standards" are along the x axis, the instrument response is plotted as the y value. No errors in the calibration "standards" are assumed.

- Don't forget to backup your data while you're working on your project! Of course here data is meant to include source code and any other files which you need for your work. It is easiest to do this routinely and therefore this is a task that can be automated with a cron-job.

- Once you've finished your project, archive your data! Ideally you will keep them in the same directory structure as your report. A suggested structure is shown in Fig. 2.5. The subdirectory "Report" contains the text of the report as well as all figures and all scripts which you used to create your figures. Of course it may make sense to have some substructure in this forlder as well. An example could be a subfolder for literature which you used and referenced in your report. The "Model" folder contains your (mathematical) model of your equipment, all subprograms and data which you generated using your model. The folder "Calibration Data" contains just that. If you had multiple calibration sets or campaigns, you will obviously want to add the corresponding subdirectories. The folder "Auxiliary Data" contains other things which you don't know where to put otherwise, such as purchase orders, invoices, data sheets, manuals, etc. If this gets too large, add substructure.

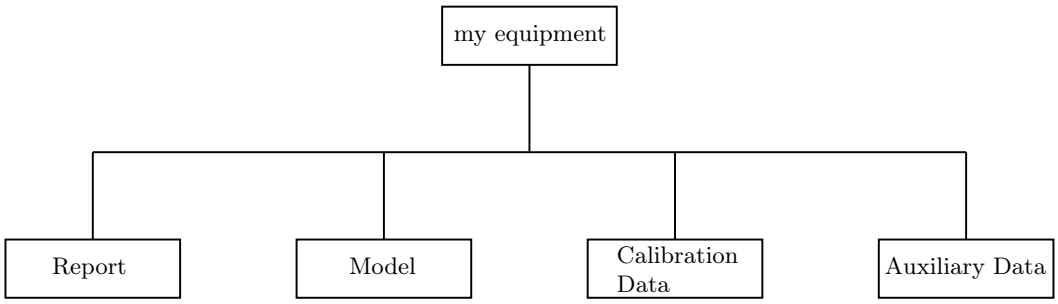


Figure 2.5: Suggested file structure for your equipment folder.

2.4 Some useful quantities

In extraterrestrial physics we have some quantities which recur in many instruments.

2.4.1 Detection Efficiency

While our particle detectors are normally designed to detect a particle with certainty or not at all, this is not possible in solar wind instruments such as PLASTIC

(*Galvin et al.*, 2008)or SWICS (*Gloeckler et al.*, 1992; *Gloeckler et al.*, 1998). A schematic of such an instrument is shown in Fig. 2.6. Basically, these instruments were designed to measure the kinetic energy, E, (or speed, v), mass, m, and charge, q, of solar wind ions. To determine these three unknowns, at least three measurements are needed: E/q, time of flight (ToF), and kinetic energy, E. The three measurements are discussed below. The trajectory of an ideal ion is shown in red in Fig. 2.6, secondary electrons in blue, critical parts of the instrument in black. These complex instruments include complicated ion- and electron optics, a time-of-flight (ToF) measurement, etc., which all conspire to reduce the detection efficiency from certainty to a smaller value. Here we generally assume that each individual process is independent of the others and that we can therefore multiply the efficiencies describing each of them. This means that each process can be carefully modeled on its own and that the final model for the detection efficiency can then be computed as a simple product. Figure 2.7 shows some of the relevant factors that need to be considered when determining the detection efficiency of a solar wind composition instrument such as SWICS or PLASTIC. Many of the factors are functions of energy and other properties of the ions that are to be measured.

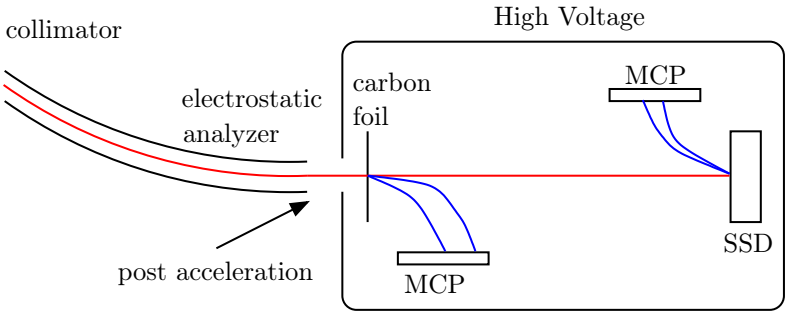


Figure 2.6: Schematic of the principal functional elements of SWICS. An ion (red) is characterized by three measurements, E/q, ToF, and residual Energy. The Time of Flight measurement is triggered by secondary electrons emitted by the thin carbon foil, its stop trigger comes from the secondary electrons from the SSD which measures residual energy.

Most ions entering SWICS will not make it through the collimator which has a very narrow angular acceptance of approximately $\pm 2°$. They are subsequently deflected by an electrostatic field which is applied between the deflection plates. Only particles within a certain E/q-band make it through this electrostatic analyzer which thus serves as an E/q-filter. Because most solar wind ions would not have enough energy to penetrate the carbon foil which is needed to provide the start trigger for the time-of-flight (ToF) measurement, they are post-accelerated

by a potential drop across the exit of the electrostatic analyzer and the carbon foil. This high voltage (between 20 and 30 kV) also requires that all signals from the following "high-voltage bubble" need to be transmitted across this high voltage. The interaction of the ion with the carbon foil is in itself a complex physical process. For SWICS the critical point is that a few secondary electrons exit the carbon foil and are accelerated towards the start micro-channel plate (MCP) by a small potential drop ($\sim$ 1 kV). It does not affect the ions very much because it is much smaller than the post-acceleration voltage, but also because the ions exit the foil with a lower charge than they had before their interaction with the foil. They fly through the "time-of-flight chamber" and hit a solid-state detector (SSD) where secondary electrons are again emitted. These too are accelerated towards an MCP to provide the stop trigger for the ToF measurement. If the ion has enough energy to trigger the SSD, its residual energy is measured. Thus each measurement in SWICS is actually the consequence of a multitude of physical processes that need to be described by a mathematical model. A possible structure of such a model is shown in Fig. 2.7.

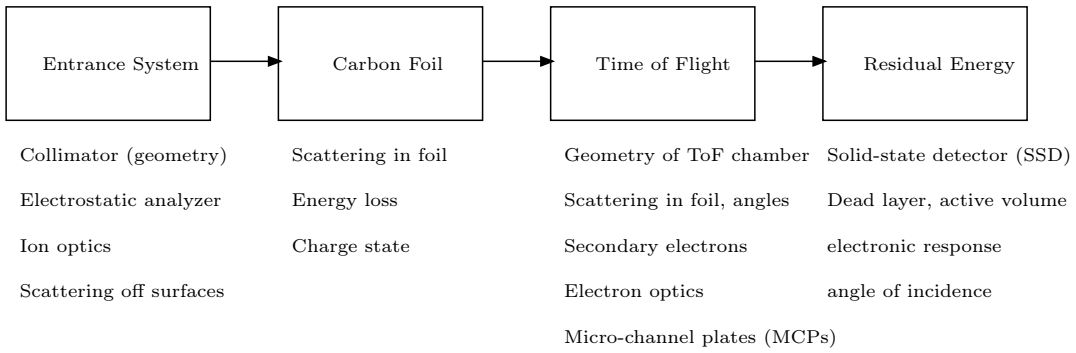


Figure 2.7: Some relevant factors for calculating the detection efficiency of a solar wind composition instrument.

Obviously, computing the detection efficiency for solar wind ions is a complex endeavoring, many PhD theses have been written about that and more information can be found in them. More complex instruments than SWICS such as PLASTIC on STEREO or HIS on Solar Orbiter (*Owen et al.*, 2020) don't only measure the speed of a solar wind ions, but its velocity vector or angle of incidence, and thus provide detailed information about the 3D velocity distribution functions of solar wind ions.

2.4.2 Geometry Factor

In its simplest form or definition, the geometry factor (or geometric factor) is a purely geometrical quantity and can be computed either analytically or numer-

ically, depending on the geometry. *Sullivan* (1971) gives a number of analytic derivations and expressions for some simple geometries. He defines the geometry factor in the following words: "For an ideal telescope - whose efficiency for detecting particles of a given type is one in a given energy interval and zero otherwise and whose sensors are mathematical surfaces with no thickness - the factor of proportionality relating the counting rate C to the intensity I is defined as the gathering power Γ of the telescope. When the intensity is isotropic, i.e., $I = I_0$, the factor of proportionality is called the geometrical factor G." That is

$$C = GI_0.$$

I highly recommend that you read this paper[1]. The concept of geometry factor is defined in Fig. 2.8. An area A can be subdivided into infinitesimal areas $\mathrm{d}A$ with their normal vectors $\mathrm{d}\vec{A}$. For an isotropic radiation field a single-element detector with an area A, the geometry factor is easily calculated,

$$G = \int_\Omega \mathrm{d}\omega \int_A \vec{r} \cdot \mathrm{d}\vec{A} = 2\pi \int \int_A \cos\vartheta \mathrm{d}A \sin\vartheta \mathrm{d}\vartheta = 2\pi A \int_0^1 \cos\vartheta \mathrm{d}\cos\vartheta = \pi A, \tag{2.1}$$

where we have used that $\mathrm{d}\cos\vartheta/\mathrm{d}\vartheta = -\sin\vartheta$ to get rid of the $\sin\vartheta$ and have implicitly taken the absolute value.

Exercise 2.1. Read the paper by *Sullivan* (1971) and calculate the geometry factor for a detector consisting of two collinear circular detectors with radii $r_1 =$ 5mm and $r_2 = 7$mm at a distance of $l = 12$mm.

A useful approximation for a quick estimate of the geometry factor of a detector consisting of two detectors with areas A_1 and $A2$ at a distance l is

$$G \leq \frac{A_1 A_2}{l^2}.$$

In the following, I provide a purely geometric derivation of the geometry factor for two collinear circular detectors. Consider an oversimplified model of a detector stack consisting of two circular detectors A and B with radii r_a and r_b at a distance l. The upper panel of Fig. 2.9 shows such an arrangement. The two detectors span a field of view with a polar angle θ such that

$$\theta = \arctan\left(\frac{r_a + r_b}{l}\right).$$

[1]There is an erratum to this paper; there were some typos which are reported in *Sullivan* (1972).

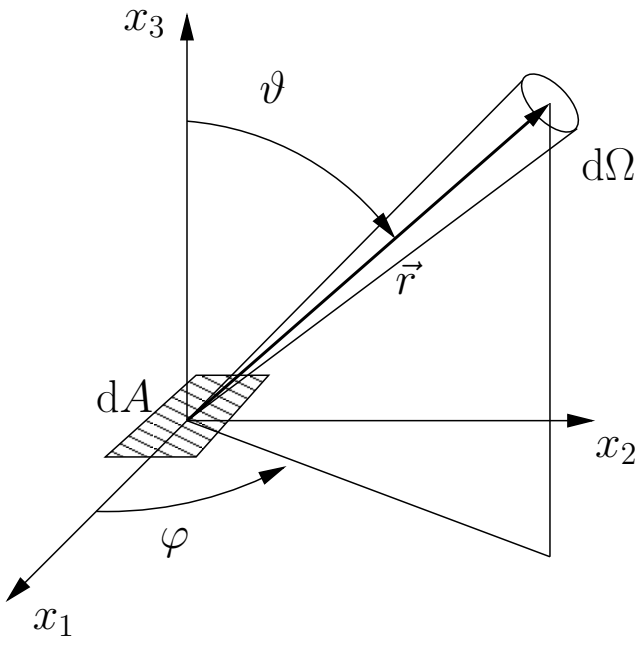


Figure 2.8: Geometry for the definition of the geometry factor.

Viewing this arrangement from an angle ζ, there will only be a small area in which both detectors are visible. Thus, as is shown in the lower panel, the determination of the geometric factor will need to consider the covering of one detector by the other. This shaded area in the lower panel can easily be calculated as

$$A_i = r_a^2 \left(\alpha_a - \frac{1}{2} \sin 2\alpha_a \right) + r_b^2 \left(\alpha_b - \frac{1}{2} \sin 2\alpha_b \right), \tag{2.2}$$

where the αs are a function of the zenith angle ζ via $d = l \tan \zeta$ and we have used $\sin 2\alpha = 2 \sin \alpha \cos \alpha$. The αs are easily found by considering the set of constituting equations for the two circles a and b,

$$\begin{aligned} x^2 + y^2 &= r_a^2, \\ (x-d)^2 + y^2 &= r_b^2, \end{aligned}$$

which solves to

$$x = \frac{r_a^2 - r_b^2 + d^2}{2d} = r_a \cos \alpha_a,$$

and

$$x' = d - \frac{r_a^2 - r_b^2 + d^2}{2d} = r_b \cos \alpha_b,$$

from which α_a and α_b follow. The effective collecting area at zenith angle ζ, $A(\zeta)$, is given by $A_i \cos \zeta$.

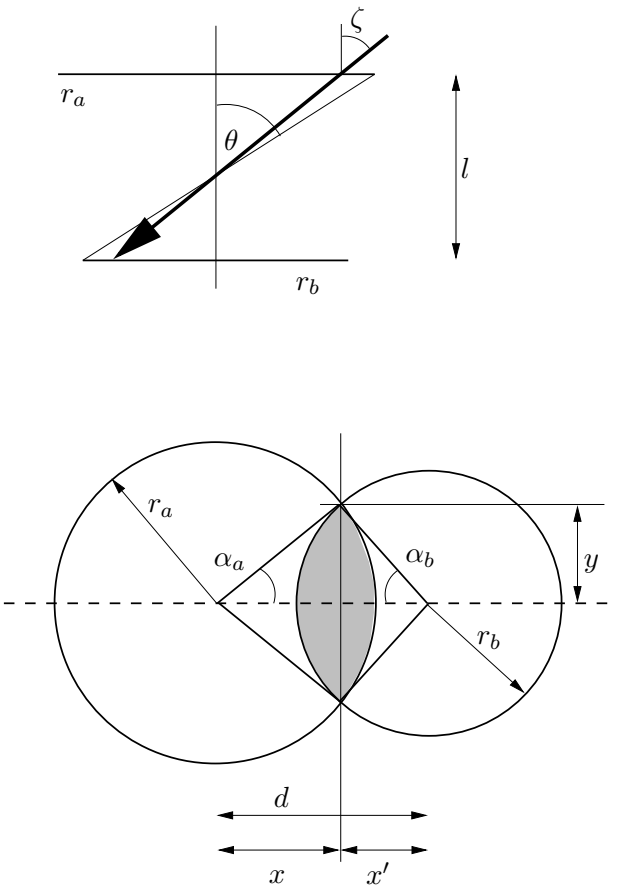


Figure 2.9: Geometry of the collinear circular detectors seen from an off-axis angle.

The geometric factor is defined by

$$G = \int \mathrm{d}A \; \mathrm{d}\Omega,$$

where Ω is the solid angle, $\mathrm{d}\Omega = \mathrm{d}\theta \sin\theta \mathrm{d}\phi$ in spherical coordinates. In this simple configuration, we have already determined $\mathrm{d}A = A_i$, as this is the area which views the solid angle $\mathrm{d}\Omega$ around viewing angle ζ. Thus,

$$\int \mathrm{d}A = A_i(\zeta),$$

and, for an isotropic distribution of incident particles we have

$$G = 2\pi \int_0^\theta A_i(\zeta) \; \mathrm{d}\zeta \;\; \sin\zeta \;\; \cos\zeta$$

for a cylindrically symmetric configuration. I did not show the φ integration because it can easily be performed, yielding the 2π infront of the integral.

Finite detector thickness: Careful inspection of the formulae given above will reveal that they are only true for infinitely thin detectors. Now particles loose energy in a detector, and they loose more if the detector is thicker. Obviously, the assumption of a zero-thickness detector is nonsense. Particles will loose a well-determined amount of energy in them, namely none. This in turn tells us that we need to consider detectors with finite thickness in our calculations. Let us do so for the simplest possible configuration of two detectors A and B with radii and thicknesses r_a, r_b and d_a, d_b at a distance l which is measured from the lower edge of the lower detector to the lower edge of the upper detector. Obviously, the geometry factor depends on the surfaces you use to calculate it.

Exercise 2.2. Calculate the effect of the finite thickness of detectors in the geometry factor.

Exercise 2.3. Particle detectors may use stacks of multiple detectors which are assembled in such a stack. The particle energy can be determined by measuring its total energy as the sum of all detected energies, provided that it stopped in one of the detectors in the detector stack. You can now design a toy model for such a detector stack with two detectors (and an imaginary "veto" detector to ensure that particles stop in the second detector) and determine the response function of this simple detector stack. For simplicity's sake, assume that the energy deposited in the detectors is proportional to its path length through the detector(s). Then the response function is given by the distribution of path lenghts through the detector. You need to consider some special cases for the geometry of this setup.

Chapter 3

Modeling as a Tool for Physical Insight

As Galilei Galileo wrote in his book "Il Saggiatore" in 1623, "The book of nature is written in the language of mathematics". While this is not an exact quote of what he wrote, it is certainly the gist of it. To spin this idea further, we may say that to understand nature, we need to develop a model and describe it in the language of mathematics. This gives us the possibility to make predictions which can be **falsified**, *the fundamental power of the scientific method.*

This chapter will not describe various models of nature as such would be impossible in any book, but is intended to point the reader to the various sources of numerical methods for some tasks which occur again and again in the mathematical description of (physical) phenomena.

While this course is more focused on statistical inference, this chapter is needed to provide the reader with tools to describe mathematical models. Only a mathematical model of a system with some free parameters allows us to find these parameters in some optimal way which is treated later on in the bulk of this book.

3.1 Solving a System of Linear Equations

Consider a system of linear equations,

$$\begin{pmatrix} a_{11} & \dots & a_{i1} & \dots & a_{n1} \\ \vdots & & \vdots & & \vdots \\ a_{1j} & \dots & a_{ij} & \dots & a_{nj} \\ \vdots & & \vdots & & \vdots \\ a_{1n} & \dots & a_{n1} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_j \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_j \\ \vdots \\ b_n \end{pmatrix}, \tag{3.1}$$

where the vector $\vec{b} = (b_1, \ldots, b_n)$ represents measurements, the matrix $\mathbf{A}$ is composed of the model matrix elements, a_{ij}, and the unknowns, x_i, are some unknown parameters or quantities in the model. If we have exact measurements and an exact knowledge of the model, then we can invert this system for the unknowns, $\vec{x}$. There are many methods to do this, *Press et al.* (1989) give an excellent summary of many of them. For most users, it will generally be good enough to use a "pre-canned" routine, for instance, `scipy` 's "solve" routine.

Example 3.1. Consider the following system of linear equations,

$$\begin{aligned} 3x + y + 5z &= 1 \\ x + 8z &= 2 \\ 2x + y + 4z &= 3 \end{aligned}$$

It is formally equivalent to the matrix equation

$$\begin{pmatrix} 3 & 1 & 5 \\ 1 & 0 & 8 \\ 2 & 1 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathbf{A} \cdot \vec{x} = (1, 2, 3).$$

In principle, all you need to do to solve this equation is to determine the inverse of the matrix $\mathbf{A}$, $\mathbf{A}^{-1}$, and multiply from the left. You would then obtain the equation

$$\mathbf{A}^{-1}\mathbf{A}\vec{x} = \mathbf{A}^{-1}\vec{b}, \quad \text{i.e.} \quad \vec{x} = \mathbf{A}^{-1}\vec{b}.$$

This may work if $\mathbf{A}$ is invertible, i.e., if the determinant of $\mathbf{A}$, det$\mathbf{A} \neq 0$, or, equivalently, the rank of $\mathbf{A}$ equals the number of columns and rows, n, in other words, if the rows in $\mathbf{A}$ are all linearly independent.

An easy way to obtain the inverse of a matrix is provided by `scipy` 's `inv` routine:

```
from numpy import *
from scipy.linalg import inv
a = array([[3,1,5],[1,0,8],[2,1,4]])
b = array([1.,2.,3.])
inva = inv(a)
print(inva)
one = dot(inva,a)
print(one) #just to check...
print('x = ', dot(inva,b))
```

It is generally better to use the following code:

```
from numpy import *
from scipy.linalg import solve
a = array([[3,1,5],[1,0,8],[2,1,4]])
b = array([1.,2.,3.])
x=solve(a,b)
print('x = ', x)
```

Note, however, that uncertainties in the matrix elements, a_{ij}, usually don't allow a straightforward inversion process to solve the system. It is nearly always preferable to solve such problems in an iterative manner (forward modeling).

Since we're already in the midst of linear algebra, let me point out that `scipy` and `numpy` provide a plethora of linear algebra tools and routines. For instance, the determinant of a matrix is easily computed:

```
from numpy import *
from scipy.linalg import det
a = array([[3,1,5],[1,0,8],[2,1,4]])
print('det(A) = ', det(a))
```

Note that `scipy.linalg` contains all of `numpy.linalg`'s functions plus some other additional ones. Useful guides are provided under the following address:
`https://docs.scipy.org/doc/scipy/`

Another example is the determination of the eigenvalues, λ, and eigenvectors, $\vec{v}$, of a matrix, $\mathbf{A}$. They are defined by

$$(\mathbf{A} - \lambda\mathbf{I})\,\vec{v} = 0,$$

where $\mathbf{I}$ is the unity matrix. They are easily found using the following code snippet:

```
from numpy import *
from scipy.linalg import eig
a = array([[3,1,5],[1,0,8],[2,1,4]])
eigen_val, eigen_vec = eig(a)
```

If the matrix has three linearly independent eigenvectors, this code snippet will result in three eigenvectors and three eigenvalues.

3.2 Numerical Integration

TODO

3.2.1 The Trapezoid Rule

3.2.2 Simpson's Rule

3.2.3 Runge-Kutta

3.3 Solving an Ordinary Differential Equation (ODE)

Dynamical systems are described by differential equations and so it is important to be able to solve such equations. Lets see how to do this with the simple example of a charged particle in a constant magnetic field, $\vec{B}$. In other words, we need to integrate the equation of motion of a particle with mass m and charge q in a constant magnetic field, $\vec{B}$. The force is given by the Lorentz force, and so we have

$$\vec{F} = m\,\vec{a} = q\,(\vec{v} \times \vec{B}) \quad \text{or} \quad \vec{a} = \frac{q}{m}\,(\vec{v} \times \vec{B}). \tag{3.2}$$

We will use `scipy` 's `odeint` routine which integrates a system of Ordinary Differential Equations (that's what the "ODE" stands for). We check how to use it: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.odeint.html`. As expected, we need to provide the function which is to be integrated (eq. 3.2), as well as the initial conditions and an array with the "times" for which the solutions are needed. Finally, we normally also need to provide some arguments to the function, for instance the charge and the mass of the particles, q and m. The solution to eq. 3.2 is obtained by calling

```
sol = odeint(func, y, t, args)
```

where `func` is the function to be integrated, `y` are the initial conditions, `t` is the array of times for which we want to know the positions of the particle, and `args` contains the arguments to the function. Of course "time" does not necessarily need to be time, it stands for the independent variable. The solution is obtained by turning the second-order ODE into two first-order ODEs by defining a new dependent variable, in this case velocity:

$$\begin{aligned} \vec{v} &\doteq z', \quad \text{with} \quad \vec{v}(0) = \vec{v}_0,\ \vec{z}(0) = \vec{z}_0, \\ \vec{v}' &= \frac{q}{m}\,(\vec{v} \times \vec{B}). \end{aligned} \tag{3.3}$$

Here I have used a prime to indicate the derivative, i.e., $v = z'$ means that v is the derivative of z with respect to the independent or controlling variable. The vector $\vec{y} = (\vec{v}, \vec{v}')$ is passed to `odeint` as the variable `y` and the function `func` needs to understand that notation:

```
def func(y, dt, q, m):
    x, v = np.hsplit(y, 2)
    xp = v
    vp = np.multiply(q/m, np.cross(v, B_field))
    z = np.hstack([xp, vp])
    return z
```

The line `x,v = np.hsplit(y,2)` calls the command `split` from numpy and splits the "vector of vectors" y into two vectors, $\vec{x}$, and $\vec{v}$ which contain the position and velocity of the particle.

We now provide the values for $\vec{B}$, q, and m:

```
B_field = [0.,0.,1.]
q = -1.
m = 1.
```

and the main part of the script:

```
def main():
    vel = [0., 1., 0.1]
    x   = [1., 0., 0.]
    t = np.linspace(0,100,2000)
    y = np.hstack([x, vel])
    z = odeint(func, y, t, args=(q, m))
    x  = z[:, 0:3]
    v  = z[:, 3:6]
    ax = plt.figure().add_subplot(projection='3d')
    ax.plot(x[:, 0], x[:, 1], x[:, 2])
    plt.show()

main()
```

We need to preceed our code with some declarations which load the packages which are called:

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

We end this section with an example which illustrates the treatment of a simple harmonic oscillator,

$$m\ddot{x} + g\dot{x} + kx = A\sin(\omega t). \tag{3.4}$$

```
from math import pi
from scipy.integrate import odeint
from numpy import *

#Define parameters
m = 1.0    #mass
k = 1.0    #spring constant
g = 0.1    #damping constant
A = 1.0    #amplitude of driving force
w = 10.*pi #pi is imported from math

#Define initial conditions
x0 = 0.0  #initial position
xp0 = 1.0 #initial speed

#Define computational parameters
nt = 101      #number of timesteps for output
t_stop = 30. #time to stop

#Define filenames
outfilename = "oscil_v1.dat"
```

As before, we solve this using the well-known trick to reduce a second-order ODE to a system of two first-order ODEs:

```
def func(y,t):
    x,v = hsplit(y,2)
    xp = v
    vp = A/m*sin(w*t) - g/m*v - k/m*x
    z = hstack([xp, vp])
    return z
```

We set up the initial conditions and write the solution to a file.

```
x = x0                          #initial position
v = xp0                         #initial speed
y = hstack([x,v])               #initial conditions
t = linspace(0., t_stop, nt)    #times at which to evaluate
z = odeint(func,y,t)
x_arr, v_arr = hsplit(z,2)

#open output file
```

```
with open(outfilename, 'w') as f:
    i = 0
    while i < nt:
        f.write('%g \t %g \n' % (t[i], x_arr[i]))
        i = i + 1
```

We can easily visualize the solution:

```
import  matplotlib.pyplot as plt
import numpy as np

x,y = np.loadtxt('oscil_v1.dat', usecols=(0,1), unpack=True)
fig, ax = plt.subplots()
ax.plot(x,y)
ax.set_xlabel('time [a.u.]')
ax.set_ylabel('amplitude [a.u.]')
plt.show()
```

Of course you are encouraged to play around with these examples by changing their initial and other defining conditions.

3.4 Solving Coupled Differential Equations: Coupled Oscillators

Often one needs to solve systems of coupled differential equations, as for instance a set of three coupled oscillators. They should be allowed to have different masses, m_1, $m2$, and m_3 and different friction coefficients, b_1, b_2, and b_3, but we only allow one coupling constand, D.

```
from scipy.integrate import odeint
from numpy import loadtxt, savetxt, hsplit
import numpy as np
import matplotlib.pyplot as plt

#Definitions of constants of the problem
#Masses:
m1 = 1.; m2 = 1.; m3 = 1.
#friction coefficients
b1 = 0.25; b2 = 0.25; b3 = 0.25
#coupling constant between oscilators
D = 1.
```

```
#Now pack all constants into a "vector"
p = [m1, m2, m3, b1, b2, b3, D]
```

We define the functions that is to are integrated. Because we have three coupled oscillators, we now have six equations instead of the two in the preceeding example of the harmonic oscillator.

```
def eqs(x,t,p):
    """
    Define the function that needs to be integrated.
    x:  Vector of x_i which describe the current state
    t:  time
    p:  parameters (m_i, b_i, D)
    """

    x1, y1, x2, y2, x3, y3 = x
    m1, m2, m3, b1, b2, b3, D = p

    #Now describe the derivatives
    f = [y1,
         (-b1*y1 - 2*D*x1 + D*x2          )/m1,
         y2,
         (-b2*y2 + D*x1 - 2*D*x2 + D*x3   )/m3,
         y3,
         (-b3*y3          + D*x2 - 2*D*x3)/m1]
    return f
```

We also need to define the initial conditions and parameters for the integration routine.

```
#Definition of the initial conditions
x1 = 0.   #All pendulums are at rest
y1 = 0.1  #Only pendulum 1 has an initial velocity
x2 = 0.
y2 = 0.
x3 = 0.
y3 = 0.

#Pack initial conditions inzo "vector" z0 (state 0)
z0 = [x1,y1,x2,y2,x3,y3]

# Define parameters for integration routine
```

```
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 25.0
numpoints = 750

# Define the time steps for which we want solutions
t = np.linspace(0, stoptime, numpoints)
```

We are now ready to perform the calculation and store the results in a file:

```
  z = odeint(eqs, z0, t, args = (p,), atol = abserr, \
             rtol = relerr)
  s1, sp1, s2, sp2, s3, sp3 = hsplit(z,6)
  savetxt('three-oscillators.dat', \
          np.c_[t, s1[:,0], sp1[:,0], s2[:,0], \
                sp2[:,0], s3[:,0], sp3[:,0]])
```

3.5 Solving Partial Differential Equations

3.5.1 PDEs of the hyperbolic, parabolic, and elliptic kinds

3.5.2 The Diffusion Equation, a Parabolic PDE

An Explicit Scheme

diff_1.py

An Implicit Scheme

diff_impl.py

3.5.3 Hyperbolic Equations

3.5.4 The Wave Equation, a Hyperbolic PDE

1dwave_simple.py

3.5.5 Elliptic Equations

3.5.6 Poisson's Equation, an Elliptic PDE

3.6 Solving Partial Differential Equations in 2 Dimensions

2dwave_simple.py
    half-circle-membrane.py

3.7 Introducing Boundary Conditions

While the partial differential equations which describe a problem may be seen to "rule the world", you should not be too gullible about that. Remember, it is the same PDEs of aerodynamics which govern the behavior of a 17-th century windmill and how an A-380 flies from Frankfurt to Beijing. The difference lies in the boundary conditions. So **never underestimate your boundary conditions!**

3.8 Fast Fourier Transforms

3.9 Wavelet Transforms

3.10 Interpolation

Chapter 4

A Reminder of Concepts in Probability

What is a probability? We might say that it is the likelihood of something happening. But what does that mean? Mathematically, we say that probability is a pair (P, Ω) of a set Ω and a probability function, P, which somehow maps the possible subsets of Ω, $\mathrm{d}\Omega$, to a real numper, $0 \leq P(\mathrm{d}\Omega) \leq 1$. Because something can not be more probable than 100% certainty, we have that $P(\Omega) = 1$.

In this "definition", Ω is the set of all possible outcomes of an experiment or measurement. This then, poses the first difficulty in applying probabilities to the measurement process. What is Ω? Which outcomes of my experiment/measurement are possible? Which outcomes can I exclude with certainty? How should I deal with uncertainties about possible outcomes?

Nevertheless, there are some mathematical points which we should remind ourselves of:

- $P(\Omega) = 1$
- $P(\emptyset) = 0$ ($\emptyset$ is the empty set. It would contain all impossible events, but since there must not be any, it is empty.)
- $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ (A or B can happen, or even both)
- $P(A \cap B) = P(A) \cdot P(B)$ (if A and B are independent)
- $P(\hat{A}) = 1 - P(A)$ (Probability of the complementary event (not A)

Exercise 4.1. Convince yourself that this is indeed the case for a simple example, such as the well trusted (?!) die, or even more complicated, several dice! Write a

simple script that keeps track of probabilities.
The script needs to define Ω, and $P(\mathrm{d}\Omega)$ for all possible $\mathrm{d}\Omega$. It is up to you to define the space of all possible events, don't keep it trivial, but also don't make it too involved. After all, you want to check that this works!

In the previous example, I asked you to write a simple script to check the previously stated points. This of course entails that you use a reliable random number generator. As you will see in Chapter 6, this is a non-trivial feat. To generate the random integers like a die, you can use the following code snippet:

```
from numpy import random
N = 3
die = random.randint(1,7,N)
print(die)
```

Exercise 4.2. You throw a red and a green die. What is the probability of getting a 6? What is the probability of getting a 6 with the red die? What is the probability of throwing a red 6 if we already know that we threw a green 6?

4.1 Conditional Probabilities

Often, we need to compute the probability that some event may occur given the knowledge that some other event has also occured. This is an often confusing situation ...

Mathematically, it can be cast into a simple equation for the conditional probability that event A may happen given that event B has happened:

$$P(A|B) \doteq \frac{P(A \cap B)}{P(B)} \Longleftrightarrow P(A \cap B) = P(A) \cdot P(B|A) = P(B) \cdot P(A|B). \quad (4.1)$$

You can now immediately see that A and B are independent if $P(A \cap B) = P(A) \cdot P(B)$.

Equation 4.1 can be better understood if we interpret it in the following way. The probability of event B to happen given that various "precursors" A_i may be needed is given by:

$$P(B) = P(A_1) \cdot P(B|A_1) + \ldots + P(A_n) \cdot P(B|A_n). \quad (4.2)$$

Formally this is the probability of event B happening (or having happened) given the pairwise disjoint events A_i occured and that $A_1 \cup \ldots \cup A_n = \Omega$. The logics behind this definition are summarized in Fig. 4.1.

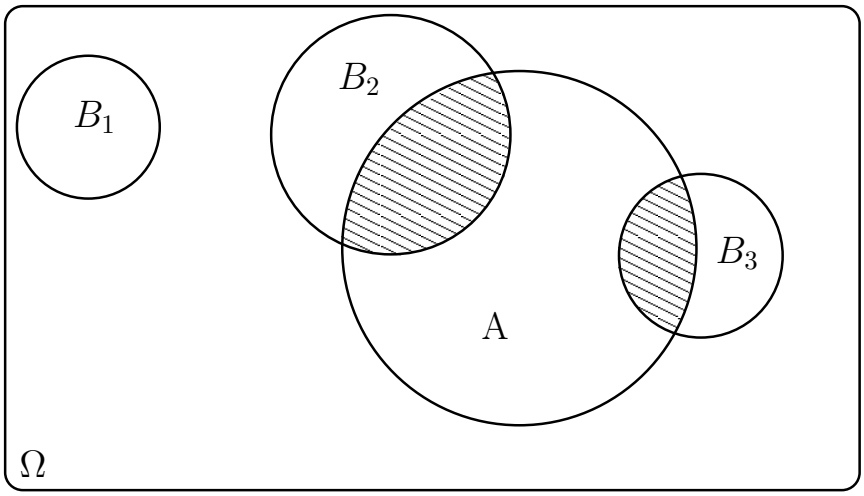


Figure 4.1: Illustration of the meaning of conditional probability. In essence, the probability of A is "renormalized" to that of B instead of all probability space Ω.

Example 4.1. The local handball team THW is about to play against the winning team of the semi-final between Flensburg and, say, Berlin. We want to compute the probability that THW will win. Obviously, this is not 1:1! We need to consider the history of past games to make an educated guess. We consider the past performance of the teams playing against each other. We give this information in the format of (team A wins agains team B):(team B wins agains team A).
Flensburg: Berlin – 3:1;
THW: Flensburg – 2:1;
THW: Berlin – 4:1.
Now using eq. 4.2 we have

$$\begin{aligned} P(\text{THW wins}) &= P(\text{Flensburg wins semi} - \text{final}) \cdot \\ &\quad P(\text{THW wins final}|\text{Flensburg wins semi} - \text{final}) \\ &\quad + P(\text{Berlin wins semi} - \text{final}) \cdot \\ &\quad P(\text{THW wins final}|\text{Berlin wins semi} - \text{final}) \\ &= 0.75 \cdot 0.67 + 0.25 \cdot 0.8 \approx 0.7 \end{aligned}$$

In other words, the probability of THW winning the final is 70% percent, which is larger than 50%.

Example 4.2. You again throw a red and a green die. Now consider the following two outcomes and compute their probabilities:
a) Both dice show the same number.
b) You already know that the sum of the numbers is 8 and now you want to know

the probability that both dice show the same number.
The probability of a) is $P(a) = 1/6$ because the first number is irrelevant and the chance of getting the same number in the second throw is again $P = 1/6$. On the other hand, if we already know that the sum of the two numbers is 8, and we now want to compute the probability that both dice showed the same number of eyes, then we need to compute $P(a|b) = P(a \cap b)/P(b)$ to obtain the conditional probability of the event described in b). The probability that the sum of the two numbers is 8 is 5/36 because of all 36 possible combinations only 5 give a sum of 8. Thus $P(b) = 5/36$. The probability that both dice show a 4 is $P(a \cap b) = 1/6 \cdot 1/6 = 1/36$, and so[1]

$$\frac{P(a \cap b)}{p(b)} = \frac{\frac{1}{6} \cdot \frac{1}{6}}{\frac{5}{6} \cdot \frac{1}{6}} = \frac{1}{5}.$$

Thus the probability of a) increases if we already know that b) is true. This gets even more dramatic if we know that the sum of numbers is 10 (and even more so if we know that the sum is 12...). Obviously, if we know that the sum is odd, ...

4.2 Combinatorics

4.2.1 Permutations

Factorial ($n!$): If you have a red, green, and blue ball, there are 6 ways of arranging them,

$$(r, g, b), (r, b, g), (g, r, b), (g, b, r), (b, r, g), (b, g, r).$$

n different and distinguishable entities can be arranged in $n!$ different ways.
The number of permutation of n different elements with n_1 indistinguishable elements of the first kind, n_2 of the second, ..., and n_k of the k-th kind is

$$P_n^{(n_1, n_2, \ldots, n_k)} = \frac{n!}{n_1! n_2! \ldots n_k!}, \quad \text{where} \quad \sum_{i=1}^{k} n_i = n.$$

4.2.2 Combinations

Binomial Coefficient ($\binom{n}{k}$): Consider bag which contains an apple (a), an organge (o), a pear (p), and a piece of bread (b). You can choose two items from

[1] We may also forumlate this differently: The probability of both dice showing the same number is $P(a) = 1/6$ as we already saw in a). The probability that the one of them shows exactly 4 is also 1/6, and so $P(a \cap b) = 1/36$.

it. Then there are $\binom{4}{2} = 6$ different combinations you can choose:

$$(ao), (ap), (ab), (op), (ob), (pb).$$

Thus, the binomial coefficient gives the number of combinations without repetitions. You have a bag with a fixed number of n items in it from which you pick a number of articles without replacing them.

The number of combinations of k objects chosen out of n objects ($n > k$) without replacements is

$$C_n^k = \binom{n}{k} = \frac{n \cdot (n-1) \cdot \ldots \cdot (n-k+1)}{k!} = \frac{n!}{k!(n-k)!}.$$

The binomial coefficient is informally called "n choose k" or "n over k". Note that

$$\binom{n}{n-k} = \binom{n}{k}.$$

It is easy to compute the first few binomial coefficients using Pascal's triangle where each number is equal to the sum of the two numbers immediately above it:

$$\begin{array}{cc}
\binom{0}{0} & 1 \\
\binom{1}{0}\ \binom{1}{1} & 1\ 1 \\
\binom{2}{0}\ \binom{2}{1}\ \binom{2}{2} & 1\ 2\ 1 \\
\binom{3}{0}\ \binom{3}{1}\ \binom{3}{2}\ \binom{3}{3} & 1\ 3\ 3\ 1 \\
\binom{4}{0}\ \binom{4}{1}\ \binom{4}{2}\ \binom{4}{3}\ \binom{4}{4} & 1\ 4\ 6\ 4\ 1 \\
\binom{5}{0}\ \binom{5}{1}\ \binom{5}{2}\ \binom{5}{3}\ \binom{5}{4}\ \binom{5}{5} & 1\ 5\ 10\ 10\ 5\ 1 \\
\binom{6}{0}\ \binom{6}{1}\ \binom{6}{2}\ \binom{6}{3}\ \binom{6}{4}\ \binom{6}{5}\ \binom{6}{6} & 1\ 6\ 15\ 20\ 15\ 6\ 1 \\
\vdots & \vdots
\end{array}$$

Multiset Coefficient ($\left(\!\binom{n}{k}\!\right)$): If you go to the baker just before he closes and have the choice of now only 4 different kinds of bread, but only have enough money to buy three breads, you have $\left(\!\binom{4}{3}\!\right) = 20$ different choices. This is different from the "normal" combination discussed above because here you can choose three breads of the same kind. Or two of the same kind and one other kind.
The multiset coefficient gives the number of combinations with repetitions. You have a bag with a fixed number of n items in it from which you pick k but putting your choice back again after each draw.

$$\left(\!\binom{n}{k}\!\right) = \binom{n+k-1}{k}.$$

Combinatorics play an important role in sampling theory, but also when generating random numbers.

4.3 Using `python` for Combinatorics

`scipy` has a number of combinatorial "function" which can be called. For instance, "N choose k", $\binom{N}{k}$ is called as follows:

```
In:   from scipy.special import comb
In:   comb(4,2)
Out:  6.0
```

or also

```
In:   from scipy.special import binom
In:   binom(4,2)
Out:  6.0
```

To call the factorial,

```
In:   from scipy.special import factorial
In:   factorial(4)
Out:  array(24.0)
```

Chapter 5

Random Variables and Distribution Functions

5.1 Discrete and Continuous Random Variables

The meaning of the word "random variable" is not necessarily what we normally consider a "random" "variable". In statistics it stands for a rule that assigns a numerical value to the possible outcomes of an experiment. The experiment is considered to be a process that will produce a variable output (in the sense of "not always the same"), i.e., it is a probabilistic process. Thus we might say that the outcome of the experiment can be described by a random variable. As much as it may hurt your feelings but the results of your experiment are considered "random variables", even if you did your utmost best to design the most outstanding measurement apparatus.

Example 5.1. Take a very simple experiment, flipping a coin. Its outcome is "heads" or "tails". The number of heads counted when the coin is flipped three times is then a random variable with four possible values, $0, 1, 2, 3$.

The previous example is an example of a ***discrete* random variable**. Discrete random variables can take on only a countable number of distinct values. If the random variable X can take on n values, then the probabilities for these n values must satisfy:

$$0 \leq p_i \leq 1 \ \forall i, \quad \text{as well as} \quad \sum_{i=1,\ldots,n} p_i = 1.$$

A ***continuous* random variable**, on the other hand, can take an infinite, non-countable number of different values.

Example 5.2. You may remember the use of analog thermometers with which one could measure temperature on a scale. Because of small variations in reading

the scale one could read different temperatures even when the thermometer was in a heat bath held at constant temperature. One could even read it from the same angle a hundred times and get ever so slightly different reading. You can generalize this to an outside thermometer which is attached to your kitchen window. You could constantly read it and note down the daily averages of the temperatures throughout a year. Note that a digital thermometer while measuring the same temperature, would provide a discrete random variable because it can't be more precise than the ADC[1] circuit it uses.

You may now argue that even an analog thermometer does not truly provide a continuous random variable because the number of mercury atoms in it is limited. While this is, in principal and from a puristic point of view, certainly true, for all practical purposes, the thermometer is a good example. Phew.

For a probability space (Ω, P) **a random variable** is a mapping X from Ω into the set of real numbers, $\mathbb{R}$. Or, more colloquially, a random variable is a function or rule that gives different values with different probabilities.

5.2 Distribution Functions

In this chapter we will discuss a number of common (statistical) distribution functions, how they arise and how they are used. Now there are various definitions of distribution functions and we will consider a distribution function in the sense of probability density function, similar to the familiar normalized velocity distributon function. Sometimes distribution functions are also interpreted as cumulative distributons functions which we will not do here.

Definition 5.1. We define a probability density function $p(x)$ by the following properties:

$$\begin{aligned} p(x)\mathrm{d}x &= \text{probability of observing variable } x \text{ within } [x, x+\mathrm{d}x]. \\ \int_{\infty}^{\infty} p(x)\mathrm{d}x &= 1 \end{aligned} \tag{5.1}$$

The **distribution function** or **probability distribution** or **probability density function** of a random variable X is a function $p : \mathbb{R} \to \mathbb{R}$ which assigns each possible outcome of the experiment a probability. This is easy to understand for the example of discrete random variables, there $p(x_i) = p_i$. In the case of a

[1] Analog to Digital Conversion (ADC)

continuous random variable, we define it as the probability that X has a value in $[x, x + \mathrm{d}x]$. Thus, the probability P for X to lie within the interval $[a, b]$ is

$$P(a \leq X \leq b) = \int_a^b p(x)\mathrm{d}x.$$

Often, one also needs the **cumulative distribution function (CDF)** $F_X(x)$ which is nothing else than the monotonically increasing sum (or integral) of the distribution function. It is a function that gives the probability that the random variable X has a value less than or equal to x. For discrete random variables, the CDF is given by the sum of probabilities up to p_i corresponding to x_i.

$$F_X(x) \doteq \int_{-\infty}^{x} p(x)\mathrm{d}x, \quad \text{or} \quad F_X(x_j) \doteq \sum_{i=1,\ldots,j} p_j.$$

Note that there are different definitions of distribution functions "out there". Some people call the cumulative distribution function the distribution function, others follow our approach. I've chosen this because it is close to our usual definition of distribution functions.

5.3 Support

In a sense, the integration limits of eq. 5.1 are an unnecessary overkill. Many probability density functions are limited to a finite domain. Have you ever seen a particle which travels faster than light? Thus, before we discuss individual distribution functions, we define the support of a real-valued function as the subset of the domain which contains all those elements of the independent variable which are not mapped to zero.

$$\mathrm{supp}(f) = \{x \in X | f(x) \neq 0\}. \tag{5.2}$$

5.4 Expectation Value and Variance

Definition 5.2. The expectation value or expected value of a discrete random variable is given by the weighted average

$$\mathbb{E}[X] = \mu = \sum_{k=1}^{N} P(X = k) \cdot k, \quad \text{where } N \text{ can be } \infty.$$

For a continuous random variable, the expectation value is given by

$$\mathbb{E}[X] = \mu = \int_{-\infty}^{\infty} p(x) \cdot x \ \mathrm{d}x,$$

where the integration limits only need to cover the support of p, $\mathrm{supp}(p)$. One often finds the Greek letter μ denoting the expectation value.

Definition 5.3. The variance of a random variable with expectation value $\mathbb{E}[X] = \mu$ is given by

$$\mathrm{Var}[X] = \mathbb{E}\left[(X-\mu)^2\right].$$

For discrete and continuous random variables, X, the variance is given by

$$\mathrm{Var}[X] = \sum_{i=1}^{N} P(X=k)\cdot(k-\mu)^2, \qquad \mathrm{Var}[X] = \int_{-\infty}^{\infty} p(x)\cdot(x-\mu)^2\,\mathrm{d}x.$$

The variance is a measure of how wide the distribution is, i.e., how much spread there is or how much variability is inherent in the distribution. We will consider some examples in a moment.

Definition 5.4. Where there is a variance, there is also a standard deviation, σ, of the random variable, and it is the square root of the variance:

$$\sigma(X) = \sqrt{\mathrm{Var}[X]}.$$

Example 5.3. Consider our good friend, the die. What is the expectation value for all the throws if it is a fair die, i.e., $P(X=k) = 1/6\ \forall k$?

$$\mathbb{E}[X] = \sum_{i=1}^{6} \frac{1+2+3+4+5+6}{6} = 3.5.$$

The variance is computed to be

$$\mathrm{Var}[X] = \sum_{i=1}^{6} \frac{1}{6}\,(k-3.5)^2 = 2.92,$$

and the standard deviation is then $\sigma = \sqrt{\mathrm{Var}[X]} = \sqrt{2.92} = 1.71$.

Example 5.4. Consider the uniform deviate on the support $[0,1]$, $f(x) = 1$ on $[0,1]$. The expectation value is

$$\mathbb{E}[X] = \int_0^1 x\mathrm{d}x = 0.5 = \mu$$

and the variance and standard deviation are

$$\mathrm{Var}[X] = \int_0^1 (x-\mu)^2\mathrm{d}x = \int_{-1/2}^{1/2} z^2\mathrm{d}z = \left[\frac{z^3}{3}\right]_{-1/2}^{1/2} = 0.083, \quad \sigma = \sqrt{0.083} = 0.289.$$

5.4.1 Some Formulae for the Expectation value and Variance

Consider two random variables, X and Y. Then the following holds true even if they are dependent:

$$\begin{aligned}
\mathbb{E}[X+Y] &= \mathbb{E}[X]+\mathbb{E}[Y], \qquad \text{Additive} \\
\mathbb{E}[aX] &= a\mathbb{E}[X], \qquad \text{Multiplicative} \\
\mathrm{Var}[X] &= \mathbb{E}\left[X^2\right]-\mathbb{E}[X]^2, \\
\mathrm{Var}[aX] &= a^2\cdot\mathrm{Var}[X] \quad \forall a\in\mathbb{R}, \\
\mathrm{Var}[X+a] &= \mathrm{Var}[X] \quad \forall a\in\mathbb{R}.
\end{aligned}$$

If X and Y are independent, then the following also holds:

$$\mathbb{E}[X\cdot Y]=\mathbb{E}[X]\cdot\mathbb{E}[Y], \qquad \mathrm{Var}[X+Y]=\mathrm{Var}[X]+\mathrm{Var}[Y].$$

5.5 Covariance and Correlation

Covariance is a property which can only be held by a pair of random variables (or more). It is a measure of how much the two random variables X and Y vary together or of their joint variance, in other words, their *co*-variance. If Y tends to be large if X is large and tends to be small if X is small, then their covariance will be positive. If X tends to be small when X is large and large when X is small, then the covariance will be negative. It is best to illustrate this with an example.

Example 5.5. Consider two dice, die 1 and die 2 which you throw 10 times. Suppose you get the following results:

$$\begin{aligned}
D_1 &: [1,6,3,4,6,4,6,5,6,5] \\
D_2 &: [5,5,5,5,6,1,3,2,6,6]
\end{aligned}$$

While there may be some covariance in this example, you don't expect one if you throw the dice often enough. After all, the two random variables D_1 and D_2 are independent. But now what happens if you consider D_1+D_2 as the two random variables? Then you *do* expect a positive covariance! This is illustrated in Fig. 5.1.

The covariance of two random variables X and Y is given by

$$\begin{aligned}
\mathrm{Cov}[X,Y] &= \mathbb{E}[(X-\mathbb{E}[X])(Y-\mathbb{E}[Y])], \\
&= \mathbb{E}[XY-X\mathbb{E}[Y]-\mathbb{E}[X]Y+\mathbb{E}[X]\mathbb{E}[Y]], \\
&= \mathbb{E}[XY]-\mathbb{E}[X]\mathbb{E}[Y]-\mathbb{E}[X]\mathbb{E}[Y]+\mathbb{E}[X]\mathbb{E}[Y], \\
&= \mathbb{E}[XY]-\mathbb{E}[X]\mathbb{E}[Y],
\end{aligned}$$

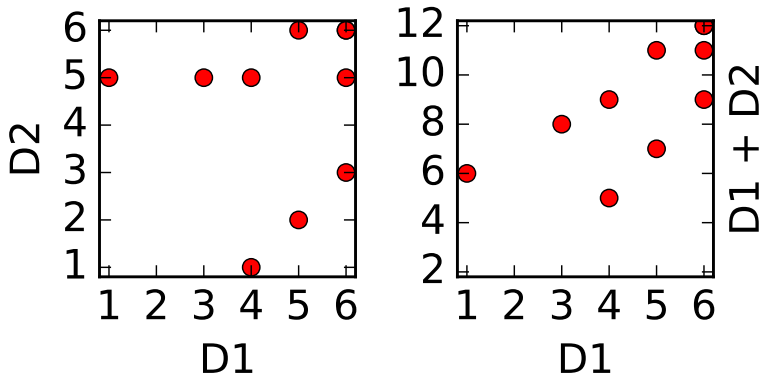


Figure 5.1: Illustration for the covariance of independent (left) and dependent (right) random variables. Two dice D_1 and D_2 are thrown 10 times. The left hand panel shows the results of die D_2 plotted vs. those of die D_1. The two random variables are independent and the covariance should vanish. The right-hand panel shows the sum of the results of the two dice $(D_1 + D_2)$ plotted vs. the results of die D_1. These two random variables are no longer independent and show covariance.

where we have used the formulae for the expectation value given on page 43. *Do not use the last line to compute the covariance!* As you may expect by closely looking at that line, it is prone to numerical cancellations and may result in non-sensical results. Use `numpy.cov` instead.

There are two more things to be said about the covariance itself:

$$\mathrm{Cov}\left[X, X\right] = \mathrm{Var}\left[X\right], \quad \text{and} \quad \mathrm{Var}\left[X + Y\right] = \mathrm{Var}\left[X\right] + \mathrm{Var}\left[Y\right] + 2 \cdot \mathrm{Cov}\left[X, Y\right].$$

If X and Y are independent the last term in the second equation vanishes.

As you may already have guessed, the covariance of two random variables X and Y is closely connected with the correlation between the two. The correlation coefficient ρ of X and Y is

$$\rho = \frac{\mathrm{Cov}\left[X, Y\right]}{\sigma(X)\sigma(Y)},$$

where $\sigma(X) = \sqrt{\mathrm{Var}\left[X\right]}$ is the standard deviation of X. Division of the covariance between X and Y by their standard deviations (if they are different from zero) normalizes the covariance to the correlation coefficient which can vary between -1 and 1.

Example 5.6. Let us illustrate this with our two dice again and consider D_1 and $D_1 + D_2$ as the random variables. We know that $\mathbb{E}\left[D_1\right] = 3.5$ and $\mathbb{E}\left[D_1 + D_2\right] = 7$. Despite the above warning, we compute the covariance as

$$\mathrm{Cov}\left[X, Y\right] = \mathbb{E}\left[X \cdot Y\right] - \mathbb{E}\left[X\right] \cdot \mathbb{E}\left[Y\right] = 27.42 - 3.5 \cdot 7 = 2.92.$$

With the standard deviations $\sigma(X) = 1.71$ and $\sigma(Y) = 2.42$ the correlation coefficient is then

$$\rho = \frac{2.92}{1.71 \cdot 2.42} = 0.71$$

which, as expected, is positive.
To calculate $\mathbb{E}[X \cdot Y]$ you need to consider all possible outcomes of throwing the dice. The procedure is illustrated with the table below. If you throw a 1 with die 1, you can have a sum $D_1 + D_2 \in [2, 3, 4, 5, 6, 7]$, if you throw a 4, you could have anything in $[5, 6, 7, 8, 9, 10]$ as the sum, etc. The sum of possible sums for each possible throw of D_1 needs to be multiplied be the value of D_1. The sums of the possible sums are $[27, 33, 39, 45, 51, 57]$ and the product with the results of D_1 are $[27, 66, 117, 180, 255, 342]$. The sum of weighted sums of sums is then 987. Since there were 36 possible combinations (with some equal sums), we need to divide 987 by 36 to obtain the expectation value $\mathbb{E}[X \cdot Y] = 27.42$.

die 1	1	2	3	4	5	6
$D_1 + D_2$	2	3	4	5	6	7
	3	4	5	6	7	8
	4	5	6	7	8	9
	5	6	7	8	9	10
	6	7	8	9	10	11
	7	8	9	10	11	12
sum	27	33	39	45	51	57

5.6 The Binomial Distribution (Again)

Consider an unfair or lopsided coin which has probability p for heads and probability $q = 1 - p$ for tails, where $p \neq q$. Now throw it n times. The probability for having thrown x heads is p^x and q^{n-x} for $n - x$ tails. The probability of the combination of x coins heads up and $n - x$ coins tails up is then $p^x q^{n-x}$. In all generality, the probability for observing x of the n flips to be in the state descriped by p is given by the binomial distribution

$$P_B(x, n, p) = \binom{n}{x} p^x q^{n-x} = \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x}.$$

The name of the binomial distribution indeed comes from the binomial theorem:

$$(p+q)^n = \sum_{x=0}^{n} \left[\binom{n}{x} p^x q^{n-x} \right].$$

The expectation value of a random variable X which is distributed according to the binomial distribution (e.g., "heads"), is given by $\mathbb{E}[X] = np$. If you flip the

(lopsided) coin n times, the expected number of heads will be np. The variance is given by $\mathrm{Var}[X] = np(1-p)$. To understand the value of the variance, consider the following. The expectation value for a single flip is p. Repeating the flip n times gives the expectation value for the number of heads as np, in other words,

$$\mathbb{E}[X] = \mathbb{E}[X_1] + \mathbb{E}[X_2] + \ldots + \mathbb{E}[X_n] = p + p + \ldots + p = np,$$

where the X_i mean a single flip of the coin. Similarly, the variance of a single flip of the coin is $\mathrm{Var}[X_i] = pq$, and so

$$\mathrm{Var}[X] = \mathrm{Var}[X_1] + \mathrm{Var}[X_2] + \ldots + \mathrm{Var}[X_n] = pq + pq + \ldots + pq = npq.$$

You may say that this begs the question: Why is $\mathrm{Var}[X_i] = pq$? Remember that

$$\mathrm{Var}[X] = \mathbb{E}\left[(X - \mathbb{E}[X])^2\right] = \mathbb{E}\left[(X^2 - 2X\mathbb{E}[X] + \mathbb{E}[X]^2)\right] = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2.$$

Now

$$\mathbb{E}\left[X_1^2\right] = \sum_x x^2 \mathrm{Prob}(X_1 = x) = 0^2 \cdot (1-p) + 1^2 \cdot p = p.$$

So

$$\mathrm{Var}[X_1] = \mathbb{E}\left[X_1^2\right] - \mathbb{E}[X_1]^2 = p - p^2 = p(1-p) = pq.$$

Binomial Distribution Summary:

$$P_B(x, n, p) = \binom{n}{x} p^x q^{n-x} = \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x}.$$

$$\mathbb{E}[X] = np, \qquad \text{and} \qquad \mathrm{Var}[X] = np(1-p).$$

5.7 The Law of Large Numbers and the Central Limit Theorem

The law of large numbers states that if you repeat an experiment a large number of times, the average result will approach the expectation value. It was first proven by Jacob Bernoulli, Poisson later named it the law of large numbers. Figure 5.2 illustrates this important law.

$$\lim_{n\to\infty} \frac{1}{n} \sum_i^n x_i = \mathbb{E}[X].$$

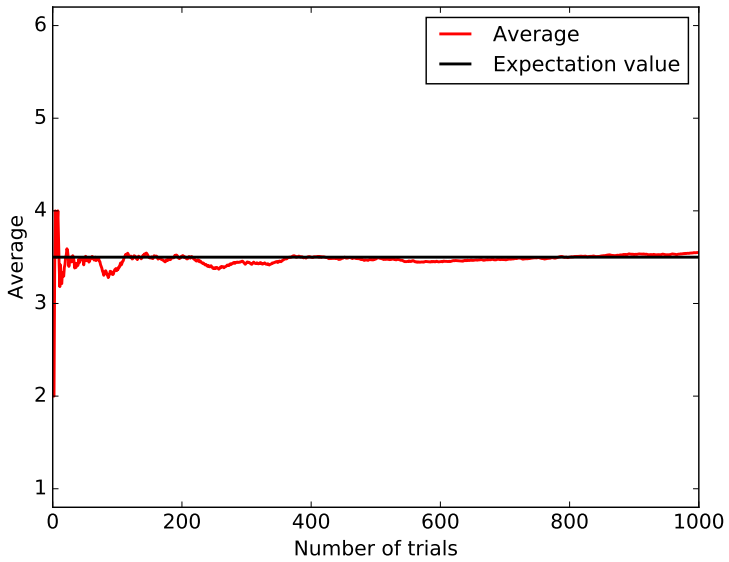


Figure 5.2: Illustration of Bernoulli's law or the law of large numbers. If we compute the average result of throwing a die, the average approaches the expectation value as the number of throws increases.

So the **law of large numbers** tells us that many repetitions of the experiment will lead to an average that tends towards the expectation value. The **central limit theorem** tells us that the results of the experiment will be distributed like a normal distribution, *even if the outcomes of the individual experiments were not normally distributed.*

Consider again our faithful die. We throw it 6 times and compute the average of the eyes. For instance, if we throw the sequence $[1, 1, 5, 3, 5, 2]$ we obtain an average of $(1+1+5+3+5+2)/6 = 17/6 = 2.833$. We now repeat this experiment 50, 200, and 2000 times, histogram the resulting averages and plot them vs. the possible averages. This is shown in Fig. 5.3.

5.8 The Normal Distribution

The central limit theorem explains why the normal distribution is so important. It is a continuous distribution, and a normally distributed random variable X can take any value in $\mathbb{R}$. One then often writes that the random variable X is normally distributed as $X \sim N(\mu, \sigma^2)$.

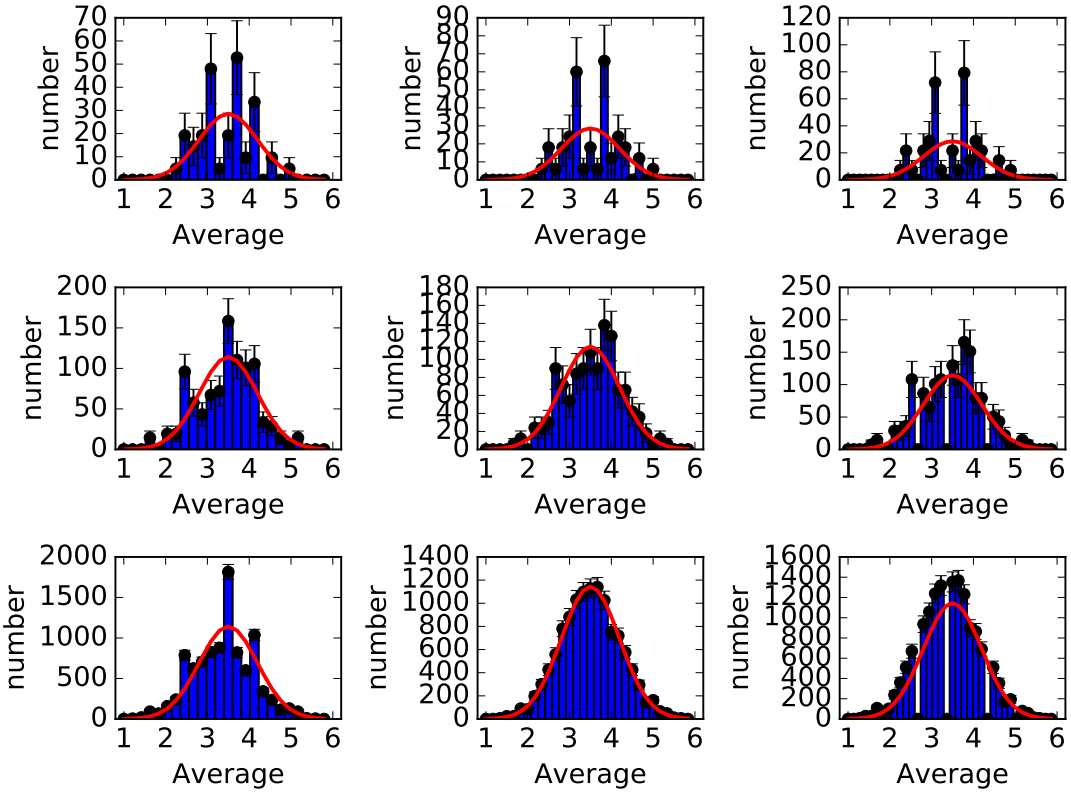


Figure 5.3: Illustration of the central limit theorem. A die is thrown 6 times and the average number of eyes recorded. That experiment is repeated 50, 200, and 2000 times (top to bottom). The average is a normally distributed random variable as is shown in the central panel. Note however, that because of the small number of throws entering the average, not all real (or even rational) values between 1 and 6 are covered. The bins need to account for that, as is shown in the left- and right-hand panels. See also section 7.9. (If you inspect this figure carefully, you will see that the "number" appears to exceed the number of tosses. This is due to the fact that one needs to plot the number of results falling within a certain bin and normalize it by the width of the bin.)

Normal Distribution Summary:

$$P_n(x,\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

$$\mathbb{E}[X] = \mu, \qquad \text{and} \qquad \mathrm{Var}[X] = \sigma^2.$$

The cumulative distribution function of the normal distribution is

$$F(x,\mu,\sigma^2) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(t-\mu)^2}{2\sigma^2}\right) \mathrm{d}t$$

and is closely related to the error function $\Phi(x)$ which for non-negative values of x gives the probability of X falling within $[-x, x]$. $\Phi(x)$ equals $F(x, \mu = 0, \sigma^2 = 1/2)$.

Rule of Thumb for the Normal Distribution: If a random variable X is normally distributed, about 68% of all values should lie within $\pm\sigma$, approximately 95% of all data points should lie within $\pm 2\sigma$, and 99.7% should lie within $\pm 3\sigma$.

A variant of the normal distribution is the standard normal distribution which is the normal distribution for $\mu = 0$ and $\sigma^2 = 1$. This distribution and its cumulative distribution function are often tabulated because one needs to look up their values quite frequently in statistics. You can also use

```
from scipy.stats import norm\\
norm.pdf(x) = exp(-x**2/2)/sqrt(2*pi) #Definition\\
norm.cdf(x) = ... #The cumulative distribution function
```

The binomial distribution function which we considered a moment ago is well approximated by the normal distribution for $np > 5$ and $nq > 5$. Both conditions need to be satisfied and of course depend on the quality of approximation which you aim at.

5.9 The Poisson Distribution

The Poisson distribution is already well-known to us because it describes radioactive decay or any stochastic events which occur sporadically with an average waiting time between events. It describes the probability of having a certain number of events in a given time interval (or spatial interval, or any other interval) if these events occur at a known constant rate and are independent of the time since the last event. The best example of a Poisson process is the activity of a long-lived

radioactive source. Another good example is the number of times a week which I need to brew coffee for our group. It is on average 4 to 5, but sometimes higher and sometimes lower. One thus says that the Poisson distribution has an expectation value, λ. This expectation value is sometimes also called intensity.

Example 5.7. On average I need to brew a fresh pot of coffee for our group 4.5 times a week. What is the probability that I need to brew coffee exactly $k = 5$ times this week?

$$P(k = 5, \lambda = 4.5) = e^{-\lambda}\frac{\lambda^k}{k!} = 17.1\%.$$

If the random variables $X_1, X_2, \ldots, X_n$ are independent Poisson random variables with λ_i being their expectation values (intensities), then the sum

$$X = X_1 + X_2 + \ldots + X_n \qquad \text{is Poisson distributed with} \qquad \lambda = \sum_{i=1,n} \lambda_i.$$

Poisson Distribution Summary:

$$P(k, \lambda) = e^{-\lambda}\frac{\lambda^k}{k!}$$

$$\mathbb{E}[X] = \lambda, \qquad \text{and} \qquad \text{Var}[X] = \lambda.$$

Rule of Thumb for the Poisson Distribution: You can approximate the Poisson distribution by a normal distribution if $\lambda \geq 9$ as is shown in Fig. 5.4.

$$P(k, \lambda) \approx \frac{1}{\sqrt{2\pi\lambda}} \exp\left(-\frac{(k-\lambda)^2}{2\lambda}\right) \quad \text{for} \quad \lambda \geq 9.$$

Note that the normal and Poisson distributions differ considerably for small numbers! If your expectation is to see half a count every second, then the normal distribution would give you a non-vanishing probability for negative counts per second! The Poisson distribution takes care of this unphysical behaviour by describing the statistics correctly.

5.10 The Weibull Distribution

The Weibull distribution models processes which have a failure rate which changes with time. The following two examples may motivate this distribution.

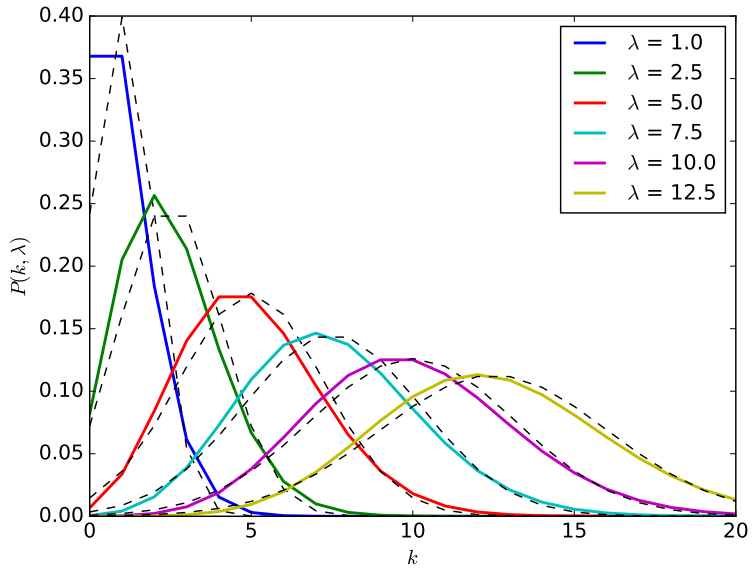


Figure 5.4: The Poisson distribution for various expectation values (or intensities), λ are shown in color. Gaussian (normal) approximations are shown as thin dashed lines.

Example 5.8. Suppose you own a company that builds coffee machines and you find that you get a number of complaints about defective machines every month. You track down every complaint and know when the coffee machine was built, assembled, and sold. You can now sort your complaints by the age of the coffee machines and check whether their failure rate is well approximated by a Poisson distribution. A Poisson distribution results from a constant probability of failure. But of course there are also other possible "failure distributions". A provider of some part in your cofee machine may have sold you some "Monday parts" even though you have a contract with them that does not allow this. This would mean that these Monday parts would fail much sooner than expected based on an average failure rate. In other words, the failure rate of your coffee machine would *decrease* with time. One calls this phenomenon an "infant mortality" of your coffee machine. Once these premature defects are "weeded out" your coffee machines show their average failure rate.

Example 5.9. Now let us assume that your company builds really good coffee machines and you have all your subcontractors under iron control. But after some years, the number of complaints begins to increase. Again you track down every complaint and plot the complaint rate against age of the coffee machine. But this time you find that the complaint rate increases with the age and use of your coffee

machines. This could be due to "the weakest joint", e.g., a moving part which sees more stress than expected in the "daily grind". It thus experiences fatigue which sets in with time. This would explain an increase of the failure rate with time.

Weibull Distribution Summary: The Weibull distribution models such non-constant failure rates as a failure rate which de- (in-) creases as a power law with time,

$$f(x,\lambda,k) = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} \exp\left(-(\frac{x}{\lambda})^k\right) & x \geq 0, \\ 0 & x < 0. \end{cases} \tag{5.3}$$

Here k is called the *shape parameter* and λ is the *scale parameter*.

If the random variable X is the mean failure rate with time (as in the examples above) and the failure rate is proportional to a power of time, then k is this power plus one. Thus we have the following interpretation:

- If $k > 1$ then the failure rate increases with time, indicating an "aging" process which results in an increased failure rate with time.
- If $k = 1$ then we have the Poisson distribution with its constant failure rate, i.e., truly random events in time.
- If $k < 1$ the failure rate decreases with time and we have a problem of "infant mortality".

In real life, a combination of "infant mortality" and an increase of failure rate with time is observed. This results in the so-called "bathtub curve" for the failure rate which is illustrated in Fig. 5.5.

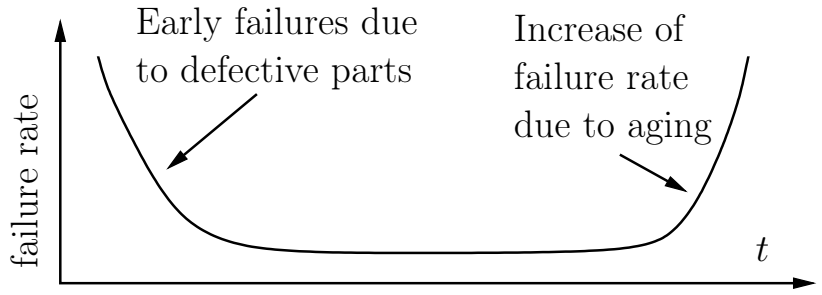


Figure 5.5: Illustration of the bathtub curve which is a well-known issue in quality assurance.

5.11 The Log-Normal Distribution

What happens to the Central Limit Theorem if you don't consider additive random variables, but ones which multiply, i.e., $\bar{X} = \prod_{i=1}^{n} X_i$? Remember that the central limit theorem says that the averages of a large number of experiments give a normal distribution around the mean. Now assume that your random variables are all positive so you can take the logarithm of the product and the resulting sum of logarithms exists because all the factors $X_i > 0$. Then the central limit theorem will ensure that the logarithms, $\log(X_i)$, are normally distributed.

Example 5.10. Say you have some money invested in stocks and one year you earn positive interest, in another year you loose a certain percentage. Now assume you have many such investments. If the percentages are normally distributed, then your investments will be log-normal distributed because the random variables, your investments, will change by factors $(1 + \varepsilon) > 0$ every time period, i.e., $\bar{X}_i = \prod_{j=1}^{n} X_{ij} = X_{i0} \prod_{j=1}^{n} (1 + \varepsilon_i)$. Then

$$\log(\frac{\bar{X}_i}{X_0}) = \sum_{i=1}^{n} \log(1 + \varepsilon_i) \approx \sum_{i=1}^{n} \varepsilon_i.$$

So the logarithms will be normally distributed.

Log-normal Distribution Summary: The log-normal distribution is the probability distribution function of a random variable whose logarithm is normally distributed.

$$p(x, \mu, \sigma) = \frac{1}{x} \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\log x - \mu)^2}{2\sigma^2}\right) \tag{5.4}$$

Here μ is still called the *mean* and σ is also still the *standard deviation.*

If Z is a standard normally distributed random variable, then

$$X = e^{\mu + \sigma Z}$$

is log-normally distributed with mean μ and standard deviation σ. Figure 5.6 illustrates the distribution (from Wikipedia).

Log-normal distributions occur quite often in nature. Many natural processes can be described by a succession of fractional growth or shrinkage, similar to the example 5.10 given above. Because they are multiplicative in nature, their results aren't described by a normal but by a log-normal distribution.

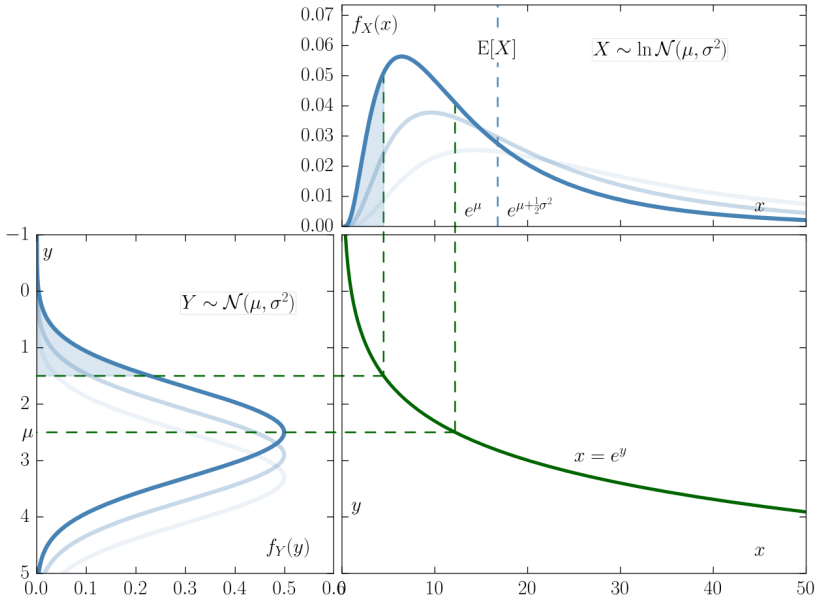


Figure 5.6: If Y is normally distributed, then X is log-normally distributed (from Wikipedia, StijnDeVuyst).

5.12 The Landau Distribution

This distribution describes the energy straggling or stochastic energy loss of particles during their passage through matter. It was found by *Landau* (1944) after whom this distribution is named.

Landau Distribution Summary: The Landau distribution is given by the following probability distribution function.

$$p(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(x + e^{-x}\right)\right). \tag{5.5}$$

The Landau distribution has such long tails that the mean and variance cannot be calculated. It is centered around zero.

5.13 The Laplace Distribution

If a number of observers measure the same quantity with the same mean μ but different normally distributed errors, then the resulting distribution of observations

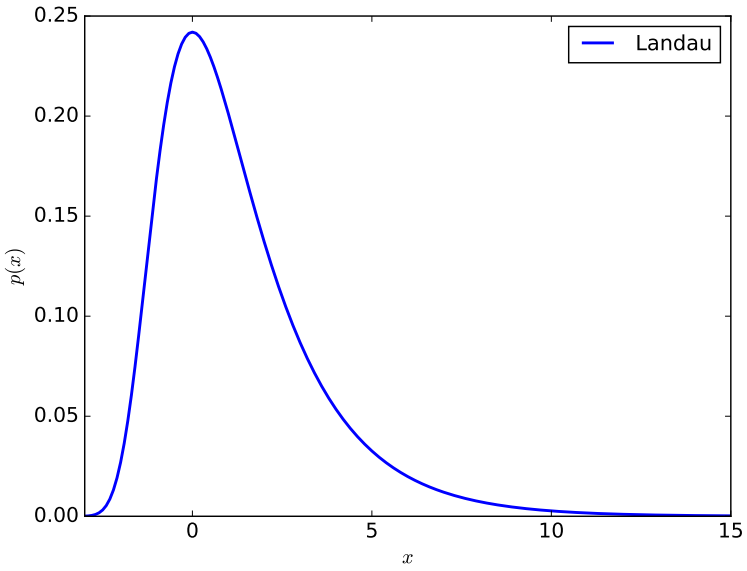


Figure 5.7: Illustration of the Landau distribution.

is well described by the Laplace distribution,

$$p(x) = \frac{1}{\sqrt{2}\sigma} \exp\left(-\sqrt{2}\frac{|x-\mu|}{\sigma}\right). \tag{5.6}$$

The Laplace probability density function (see Fig. 5.8) is maximized when μ is the median of the underlying distribution. It is useful when the underlying distribution is very heterogeneous and maybe is composed of a number of different distributions with different widths (variances). It can be used to fit regression models which make use of the median instead of the mean (see chapter 9 on fitting models to data).

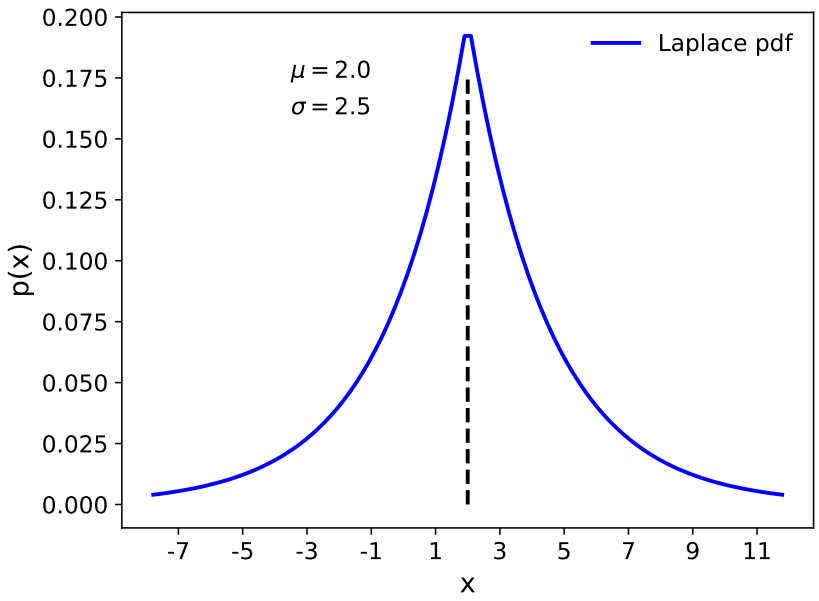


Figure 5.8: Illustration of the Laplace distribution.

Chapter 6

Generating Random Numbers

Today, generating random numbers appears trivial. `numpy` provides a collection of well-tested and highly-performant random number generators (see `numpy.random`). Nevertheless, a good understanding of how random number generators work is useful, a highly readable and recommended source is *Press et al.* (1989, and later versions) which we follow here.

6.1 Generating Uniform Random Numbers

The simplest artificial generator for discrete random numbers is probably a coin which is flipped, the next complicated generator is a die with its six sides. Both generate uniform random numbers, the first in the space $(0, 1)$, the second in $(0, 1, 2, 3, 4, 5)$ (or in $(1, 2, 3, 4, 5, 6)$ if you prefer) – if they are unbiased.

Generating random numbers with your computer is an art, and this is the reason why I leave it up to `numpy.random` to generate random numbers. There are many pitfalls in generating your own random number, the easiest way to avoid them is to use good and well-tested random number generators.

So let's generate a numerical "coin" which returns 0 or 1 for head or tails.

```
from numpy import random
coin = {0: 'heads',1:'tails'}
coin[random.randint(2)]
```

A die is generated exactly the same way, except for the argument of `randint()`.

```
from numpy import random
random.randint(1,7)
```

It is always a good idea to test whether your random number generator produces random numbers according to your expectations. For the previous example, this is trivial:

```
import numpy as np
from numpy import random
import matplotlib.pyplot as plt

rn = random.randint(1,7,6000)
bins = np.array([0,1,2,3,4,5,6,7])
counts, bins = np.histogram(rn,bins)
fig, ax = plt.subplots()
ax.set_xlabel('eyes',fontsize=14)
ax.set_ylabel('counts',fontsize=14)
x = [0.75,6.25]; yp = [1000+np.sqrt(1000.),1000+np.sqrt(1000.)]
ym = [1000-np.sqrt(1000.),1000-np.sqrt(1000.)]
ax.fill_between(x,yp,ym,facecolor='grey',alpha=0.5)
ax.plot([0.75,6.25],[1000,1000],'k-',lw=2,label='expected')
ax.set_xlim([0.5,6.5])
ax.errorbar(delete(bins,0)-1,counts,yerr=np.sqrt(1.*counts),
    fmt='bo',lw=2,label='counts')
ax.legend(loc='center right')
plt.savefig('counts.pdf')
plt.show()
```

The results are shown in Fig. 6.1. As expected when checking against deviations by one standard deviation, about one third (i.e., two) of the data points lie outside the error band. So we found nothing obviously wrong with this random generator. Of course, there are much more sophisticated tests for random number generators. If you really need to be sure about the randomness of the generator you are using, you had better run these tests. References to some tests are given in *Press et al.* (1989) which you can also find on the web (`http://numerical.recipes/`). There are also more modern references, but in principle, they all do the same, they test the null hypothesis H_0 that the random numbers generated are indeed random against the alternative hypothesis H_1 that they aren't. See Chapter 8 on hypothesis testing.

To generate uniformly distributed random numbers in an interval, use `random.rand(`$d_0, d_1, \ldots, d_n$`)` which generates an array of random numbers in the shape given by $d_0, d_1, \ldots, d_n$ drawn from the half-open interval $[0, 1)$. `numpy.random` provides many more convenience functions.

If you want random numbers in a different range, e.g., (a, b), then use `random.random_sample()` and multiply it by $(b - a)$ and add a.

```
from numpy import *
5 * np.random.random_sample((3, 2)) - 5
array([[-3.99149989, -0.52338984],
```

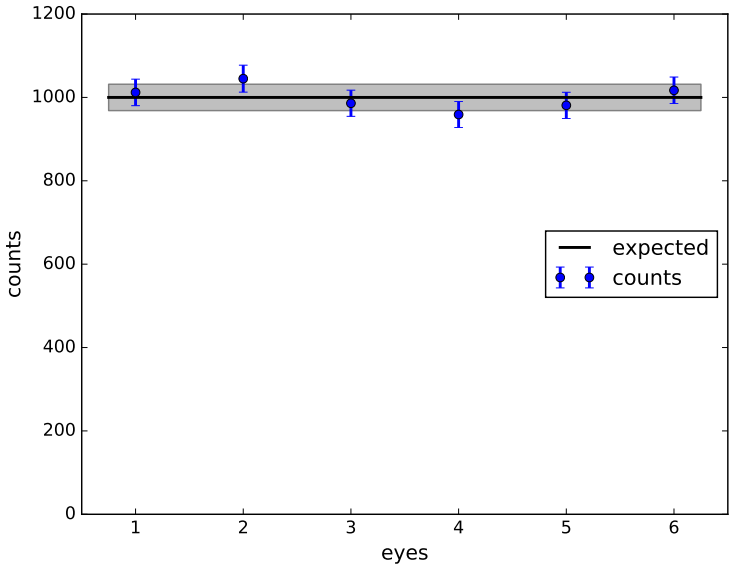


Figure 6.1: Expected (black) and simulated counts of the 6 possible outcomes of casting this "virtual die".

```
       [-2.99091858,  -0.79479508],
       [-1.23204345,  -1.75224494]])
```

6.2 The Transformation Method for Exponential and Normal Random Numbers

We don't always want uniformly sampled random number, but random numbers from a different distribution, e.g., a normal distribution or an exponential distribution (e.g., to simulate nuclear decay). This is actually not that difficult to do. On the one hand, `numpy.random` provides a wide range of random number generators. I counted more than 25 of them in March 2022.

See `https://numpy.org/doc/stable/reference/random/generator.html`.

Despite this large number of ready-made generators, I will show you how to make your own so you can generate random numbers from any distribution you like. We just saw how to generate random numbers in the range $[0, 1)$. The probability of drawing a number between x and $x + \mathrm{d}x$ is given by

$$p(x)\mathrm{d}x = \begin{cases} \mathrm{d}x & 0 \leq x < 1 \\ 0 & \text{otherwise,} \end{cases} \tag{6.1}$$

where the probability distribution $p(x)$ is normalized to unity,

$$\int_{-\infty}^{+\infty} p(x)\mathrm{d}x = 1$$

and the integral actually only needs to extend from 0 to 1. Given a uniform deviate x, we can perform some function $y(x)$ on it, which results in a new probability distribution $p(y)\mathrm{d}y$. It is easy to find out what $p(y)$ looks like:

$$\begin{aligned} |p(y)\mathrm{d}y| &= |p(x)\mathrm{d}x| \\ p(y) &= p(x)|\frac{\mathrm{d}x}{\mathrm{d}y}| = |\frac{\mathrm{d}x}{\mathrm{d}y}|, \end{aligned} \tag{6.2}$$

because $p(x)\mathrm{d}x$ is uniform across the interval $[0, 1)$. Note that we need to know $x(y)$ so we can compute $\mathrm{d}x/\mathrm{d}y$.

Now eq. 6.2 is nothing else that a differential equation for $p(y)$. Furthermore, the integral of $p(y)$ has to be unity. The solution of this differential equation is simply $x = P(y)$ where $P(y)$ is the indefinite integral of $p(y)$. Thus the transformation from the uniform deviate x into one distributed as p(y) is

$$y(x) = P^{-1}(x), \tag{6.3}$$

where $P^{-1}(x)$ is the inverse function of P. So the recipe to generate a random deviate y distributed as $p(y)$ is the following:

1. Determine the indefinite integral of $p(y)$, $P(y)$.
2. Determine its inverse, $P^{-1}(y)$
3. $y(x) = P^{-1}(x)$ is now the new deviate.

This procedure is visualized in Fig. 6.2. Because the new distribution is normalized to unity, its definite integral is unity. If we plot the indefinite integral, $P(y)$, as a function of y, the y axis will extend from 0 to 1, and the x axis from $y_{\min}$ to $y_{\max}$. Now the uniform deviate x is plotted along the y axis. If we choose a value for x (on the y axis!), say, $x = 0.7$ in Fig. 6.2, then we can transform x to $y(x)$ by intersecting with $P(y)$ and dropping down to the x axis, along which the transformed deviate $y(x)$ is plotted.

Following *Press et al.* (1989), we derive the probability distribution for exponential deviates. We choose $y(x) = -\ln(x)$, then $p(y)$ is given by

$$p(y)\mathrm{d}y = p(x)|\frac{\mathrm{d}x}{\mathrm{d}y}|\mathrm{d}y = e^{-y}\mathrm{d}y \tag{6.4}$$

because $p(x)\mathrm{d}x$ is uniform across the interval $[0, 1)$

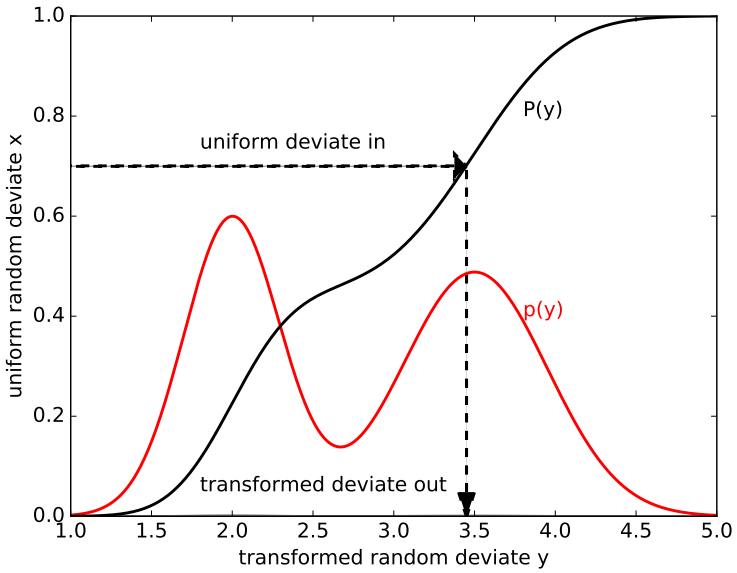


Figure 6.2: Illustration for the transformation method. The uniform deviate is plotted along the y axis, the transformed random deviate is plotted along the x axis.

6.3 Generating Random Numbers From a 'Crazy' Distribution

Sometimes one wishes to generate random numbers from a given distribution which can not be described by a function. If we have a list of pairs of values, (x, y), describing this distribution, we can read it from a file and generate random numbers in exactly the same way as described above. The following code snippet shows an example how one can provide a series of random numbers which are distributed according to a provided input (in this case the file 'some_crazy_data.dat' which is given in Tab. 6.1).

```
from scipy.interpolate import interp1d
x,y = loadtxt('some_crazy_data.dat',unpack=True)
cs_distrib = cumsum(y)
cs = cs_distrib[-1]
cs_distrib_norm = divide(cs_distrib,cs)
ran_distrib = interp1d(cs_distrib_norm,x)
```

Fig. 6.3 shows an example.

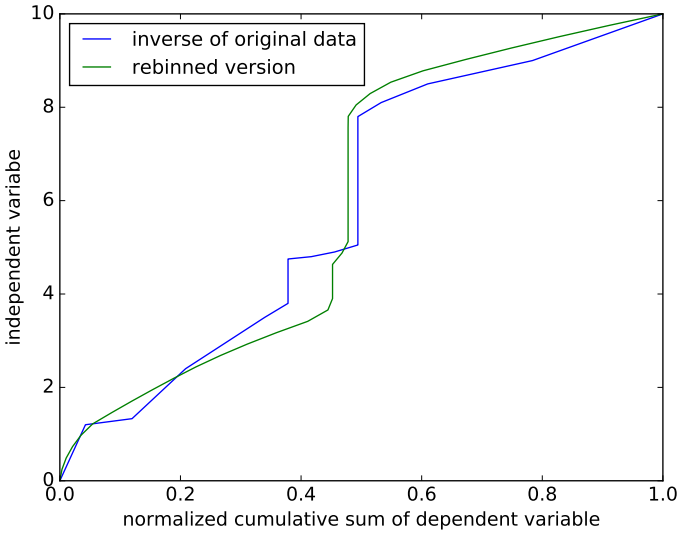


Figure 6.3: random

Exercise 6.1. Provide a random number generator with the following properties:

$$p(x) = \text{const.} \quad \text{for} \quad 0.5 \leq x \leq 1.0 \quad \text{and} \quad p(x) = \exp(-(x-1)/\lambda) \quad \text{for} \quad x \geq 1.,$$

where $p(x) = 0$ otherwise.

6.4 Generating Random Numbers on a Sphere (or on a Cirlce)

Sometimes we need to generate random locations on a sphere (hang on, you'll soon find out why). For this, we can use a nice property of the Gaussian distribution,

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2}),$$

where we have used a normal distribution with standard deviation $\sigma = 1$. A random variable that is thus distributed is said to be distributed according to $N(0, 1)$. Here the N stands for the normal distribution, the 0 for the mean or average of the distribution, and the 1 for the standard deviation. This can easily be generalized to d dimensions,

$$f(x) = \frac{1}{(\sqrt{2\pi})^d} \exp(-\frac{x \cdot x}{2}),$$

x	y
0.	0.
1.2	0.11
1.33	0.2
2.4	0.23
3.5	0.34
3.8	0.1
4.0	0.
4.6	0.
4.7	0.
4.75	0.
4.8	0.1
4.9	0.1
5.05	0.1
5.25	0.
5.6	0.
6.7	0.
7.8	0.
8.1	0.1
8.5	0.2
9.0	0.45
10.	0.56

Table 6.1: "Data" for some_crazy_data.dat.

where $x \cdot x$ denotes the scalar product in d dimensions.

Now consider a d-dimensional random vector which is distributed according to $N(0,1)$. If we rotate it by some arbitrary angle around an arbitrary axis it will still be a $N(0,1)$-distributed random vector. Every rotation can be described by a rotation matrix, U, which is always orthogonal, i.e., $UU^t = U^tU = I$, where I is the unit or identity matrix and the t indicates the transpose. Now consider a random vector $\vec{x}$ and its transform under U, $\vec{y} = U\vec{x}$. Then

$$f(y) = \frac{1}{(\sqrt{2\pi})^d} \exp(-\frac{\vec{y}\cdot\vec{y}}{2}) = \frac{1}{(\sqrt{2\pi})^d} \exp(-\frac{(U\vec{x})\cdot(U^t\vec{x})}{2}) = \frac{1}{(\sqrt{2\pi})^d} \exp(-\frac{\vec{x}\cdot\vec{x}}{2}),$$

which proves that normally distributed random numbers in d dimensions are invariant under rotations. Thus if we generate $\vec{x}$ in d dimensions and project them onto the sphere, then they are uniformly distributed on the sphere. For $d = 3$ we have a sphere, for $d = 2$ we have a circle.

```
import numpy as np

#fix random number seed so we can reproduce plots.
#Remove if you want real random numbers!
np.random.seed(20210822)

n = 500 #number of random numbers

#The following lines generate random points
#on a sphere with radius R
R = 2.
u = np.random.normal(0,1,n)
v = np.random.normal(0,1,n)
w = np.random.normal(0,1,n)
norm = (u*u + v*v + w*w)**(0.5)/R
(x,y,z) = (u,v,w)/norm
```

You can check that these points lie on a sphere and appear to be randomly distributed by plotting them using the following code snippet.

```
from mpl_toolkits import mplot3d
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.scatter(x,y,z)
plt.show()
```

6.5 Generating Random Numbers inside a Circle (or inside a Sphere)

In the previous section, we generated random points on the surface of a sphere (or on a circle) by projecting the normally distributed random points on to the boundary of the sphere (or circle). To fill a sphere or a circle, we need to adapt the radial mapping so that the points are randomly distributed inside the object. For a circle, the probability density function (PDF, see eq. 5.1 for the definition) of random points increases linearly with radius, for a sphere it increases quadratically with radius. The cumulative distribution function (CDF) thus grows quadratically (cubically). Thus we need to take the square root (cubic root) of a uniformly distributed random number as the radius of the point in the circle (sphere), as we discussed in Section 6.2. The code snippet for random points inside a circle of radius R is given below.

```
u = np.random.normal(0,1,n)
v = np.random.normal(0,1,n)
norm2 = (u*u + v*v)**(0.5)
r = np.random.uniform(0,R,n)**(0.5)
(x2,y2) = r*(u,v)/norm2
```

Again, you can check this visually by plotting:

```
ax.scatter(x2,y2)
ax.set_aspect('equal')
plt.show()
```

6.6 Building a Particle Source for an Extended Detector

We often need to simulate a detector with a finite extent in an isotropic radiation field. The problem is sketched in Fig. 6.4. It is easy to solve for a point-like detector, you simply simulate an isotropic radiation field as we did in Sec. 6.4. In that case you use generate random points on a sphere (or hemisphere), as sketched in the upper left of Fig. 6.4. But what do you do when you have an extended detector such as the blue detector in the upper right of Fig. 6.4? The solution is actually surprisingly simple. You combine the two methods of sections 6.4 and 6.5 as sketched in the bottom of Fig. 6.4. You need to take care that the circular source (shown in red) is of at least the same size as the detector (shown in blue). Then you emit radiation parallel to the radial direction which connects the center of the source to the center of the detector.

Thus the recipe is to generate a unidirectional source which is of (at least) the same size as the detector and place it at a sufficient number of random positions on a spherical source surface.

Of course, there are other solutions to this problem. A more efficient one is to start at a random point on the detector and then generate an isotropic field from a hemisphere from that point. Then you iterate across the detector.

A less efficient, but easy to implement, solution is to distribute the inward-pointing rays according to a cosine-distribution with respect to the radial direction. To understand why that also works, we need to remember that the potential inside a hollow sphere vanishes (for a homogeneous thickness of the shell). That is equivalent to the statement that from every point inside a hollow sphere you see the same surface inside a given solid angle. If you have already distributed points randomly on a sphere that means that you will always see the same number

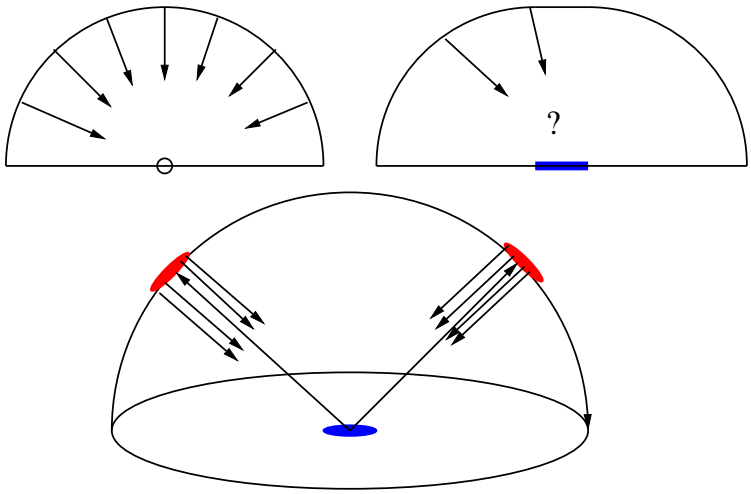


Figure 6.4: Illustration how to generate an isotropic radiation field for an extended detector (shown in blue).

(within counting statistics) of points in a given solid angle, independent of which direction is points towards. This situation is summarized in Fig. 6.5.

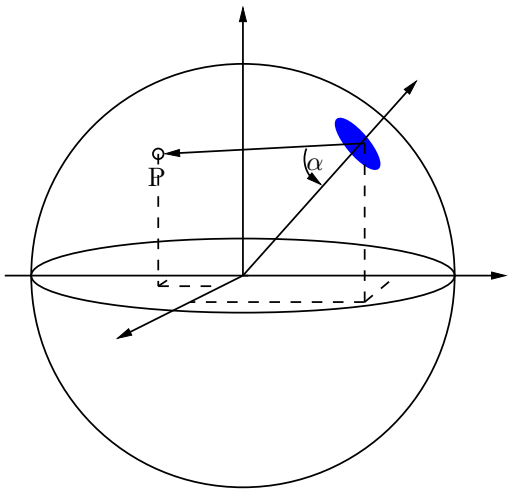


Figure 6.5: Illustration for an isotropic radiation field in a sphere.

Chapter 7

Descriptive Statistics

This chapter makes a giant leap from distributions with their known (if they exist) expectation values and variances or standard distributions to estimations of these.

> The key point to remember is that we never know the "true" expectation value and variance of the distribution function underlying a collection of measurements. We can only do our best to accurately and precisely estimate these quantities.

Example 7.1. Every car has a special on-off lighting system that allows other drivers and pedestrians to observe the driver's intentions regarding direction and speed of travel – if it is used. It is called a blinker. Interestingly, one observes that not all car drivers know about the presence of this useful property in their car.
What is the fraction of German car drivers who don't blink when changing lanes or turning right?
This is a question that is hard to answer. You can't simply check every single car and car driver all the time. The best you can do is to pick a carefully chosen random sample of car drivers and observe their habits. Therefore, you will never know the exact fraction of German car drivers who don't blink when changing lanes or turning right, but you will obtain an estimate of this fraction.

In the following we will always assume that we have n measurements or observations, $x_1, x_2, \ldots, x_n$, of a random variable, X, or of random variables, X_i. We will also use θ as the general name for parameters in our models.

Note that there is a difference between X and x: I will always write the random variable X as a capital X, and the measurement as a small x. The random variable X is the set of all possible values whereas the measurement x is the realization of an "experiment", i.e., it is one of the (possibly many) possible values, X.

Definition 7.1. The range of possible values of the parameter, θ is called the parameter space, Ω.

Definition 7.2. The function of X_i that is used to estimate θ, i.e., the statistic $u(X_i)$, is called a **point estimator** of θ.

Definition 7.3. The function of x_i that is used to estimate θ, i.e., the statistic $u(x_i)$, is called a **point estimate** of θ.

Note that an estimator acts on the random variable and the estimate uses the measurements.

7.1 The Mean, the Mode, and the Median

Definition 7.4. The **mean** or **average** of a collection of n measurements is defined as

$$\bar{x} \doteq \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{7.1}$$

It is a **point estimate** for the unknown "true" mean of the full population from which the sample was drawn.

Another useful estimator for some kind of "central value" of a population is the **most probable value** or **mode** of the distribution or population. The problem with it is that you need to bin your data which results in an uncertainty due to the binning. Alternatively you can sort your data and use a sliding window to count the number of measurements in that sliding window. The maximum is then close to the most probable value. For instance in the case of the Landau distribution described in chapter 5 the most probable value is the only "central" estimator that can be computed.

Often the distribution of measurements has some outliers which can severely influence the mean and it turns out that this is then not necessarily the best estimator for the expectation value or some other "central" value of the distributions. In that case it is often preferable to use the **median** of the distribution which is more **robust**, i.e., not as strongly influenced by outliers. Figure 7.1 summarizes the estimators for a "central value".

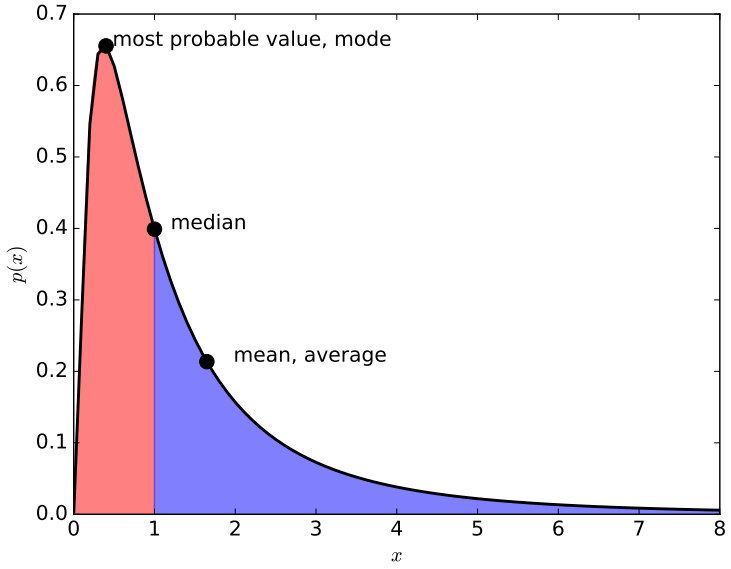


Figure 7.1: Illustration that the various estimators of the "central value" of a population with a given probability density function can lie wildly apart. The red and blue areas are the same and equal to 50% of the total area under the curve.

Definition 7.5. The **median** of a sample is defined by

$$\text{median} \doteq \begin{cases} x_{(n+1)/2} & \text{if } n \text{ is odd} \\ (x_{n/2} + x_{n/2+1})/2 & \text{if } n \text{ is even.} \end{cases} \tag{7.2}$$

Example 7.2. I have simulated a data set which is drawn from the sum of two normally distributed random variables with the same mean but a different variance. It is plotted as a black line in Fig. 7.2. In one run I generated 5 random samples from this distribution and computed the mean and median. I repeated this 1000 times and created histograms of the computed means and medians. These are also shown in Fig. 7.2 as red (mean) and blue (median) lines. Obviously, the median is more peaked than the mean and also more peaked than the original distribution. The latter is not surprising. We have seen how to generate random numbers in chapter 6.

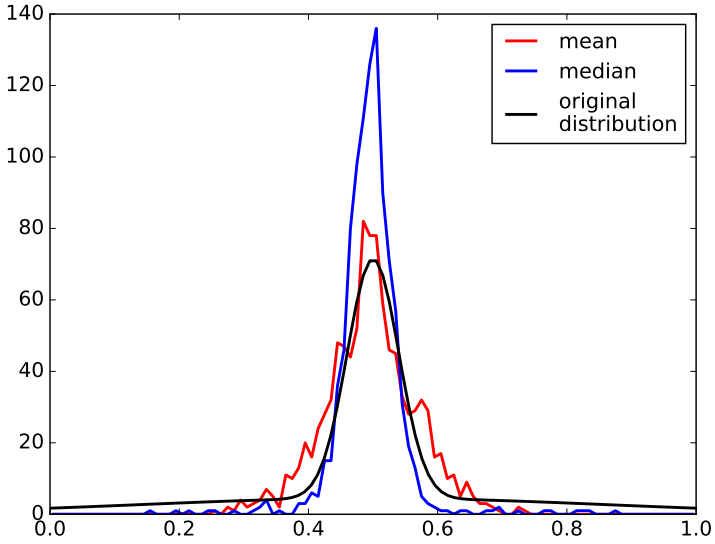


Figure 7.2: The median (blue) is a more robust estimate of the "central value" of the original distribution (black) than the mean (red). See example 7.2 for an explanation.

The study of robustness of estimators is a relatively new subject in statistics. Robust statistics attempt to estimate key properties of a population while not being affected by outliers or other assumptions underlying the analysis. Normality of the data points (i.e., a normal distribution) is often assumed but not necessarily the case. As soon as there are outliers in the data, ordinary statistical concepts may fail miserably... See section 7.7 for more details.

7.2 Empirical Variance and Empirical Standard Deviation

Definition 7.6. The empirical variance, s^2 is defined as

$$s^2 \doteq \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2. \tag{7.3}$$

Definition 7.7. The empirical standard deviation is the square root of the em-

pirical variance,

$$s \doteq \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}. \tag{7.4}$$

Note that it is the mean that enters these definitions, and only the mean!

The empirical variance is an estimate for the spread of the data (measurements), the empirical standard deviation is a measure for the spread of the observations around the mean. The reason for the $n-1$ in the denominator is explained in section 7.6.

`scipy` has a collection of useful utilities to compute the mean, median, variance, and higher moments of a sample.

```
In:  from scipy import stats
In:  from numpy import *
In:  sample = array([0,1,4,2,7,3,3,6,2,0,8,4,9])
In:  stats.describe(sample)
Out: DescribeResult(nobs=13, minmax=(0, 9),
     mean=3.76923076923076, variance=8.69230769230769,
     skewness=0.417557727193064, kurtosis=-0.98607304148066)
```

Obviously, it is up to you to cull the output to a significant number of digits. Skewness and kurtosis are described farther down in sec. 7.3. The median can be called from `numpy` :

```
In:  from numpy import *
In:  sample = array([0,1,4,2,7,3,3,6,2,0,8,4,9])
In:  median(sample)
Out: 3.0
```

7.3 Higher Moments of the Distribution

Of course, there are also higher moments of a distribution (function). Just as in physics, their use becomes more and more obscure with increasing order. Formally, they are derived from the usual moment-generating equations,

$$\text{moment}^n \doteq C\int x^n p(x)\mathrm{d}x \quad \text{or} \quad \text{moment}^n \doteq C\sum x_i^n p(x_i),$$

where C is some normalization constant.

Definition 7.8. The **skewness** of a sample is defined as

$$\text{skew}(x_1, x_2, \ldots, x_n) \doteq \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{\sigma} \right)^3 , \tag{7.5}$$

where the $\sigma = \sigma(x_1, x_2, \ldots, x_n$ is the sample's standard deviation.

A positive skewness means that the sample has an asymmetric tail towards higher values of x, a negatively skewed distribution has one extending towards lower x than the central peak of the distributions. `scipy` provides a routine to calculate the skew of a sample.

```
In:  from scipy.stats import skew
In:  from numpy.random import randn
In:  a = randn(25)
In:  skew(a)
Out: -0.22456310574520208
```

Now this is weird! We drew a normally distributed random sample from an underlying symmetric distribution and we obtained a negative skewness! It turns out that any sample, even if drawn from a perfectly symmetric distribution, will have a non-vanishing skew. The skewness defined in eq. 7.5 has to exceed a critical value to be "real" or believable. This "critical value" depends on the shape of the underlying distribution and especially its tails. For a normal distribution, this value is approximately $\sqrt{6/n}$ (*Press et al.*, 1989). In the example above skew(a) $= -0.225$ whereas $\sqrt{6/25} = 0.499$, so the returned skewness should not be trusted. As these authors state, "In real life it is good practice to believe in skewness only when they are several or many times as large as this."

Again, `scipy` provides a test for skewness,

```
In:  from scipy.stats import skewtest
In:  from numpy.random import randn
In:  a = randn(25)
In:  skewtest(a)
Out: SkewtestResult(statistic=-0.54504429743656879, \\
     pvalue=0.58572305162445459),
```

where the statistic is the computed z-score for the test and the pvalue is the 2-sided p-value for the hypothesis test. We will treat these quantities in more detail in chapter 8, but give a short summary here. The z-score is the number of standard deviations the result is away from the mean. In our case $\text{skew}(a)/\sqrt{6/25} \approx 0.5$ which is approximately the z-score. The difference is the exact value of the critical value. In other words, our result is only about half a standard deviation

away from the mean which makes it a pretty probable result. That's the second output, the p-value. It is $p \approx 0.6$ which is high. It gives the probability of a random sample $x_1, x_2, \ldots, x_n$ drawn from a normal distribution to show a skewness, $\mathrm{skew}(x_1, x_2, \ldots, x_n)$, equal to or greater than this absolute magnitude, i.e., $|\mathrm{skew}(a)|$. In chapter 8 we will learn how to decide whether to trust the result of a (skewness) test.

The next higher moment of a sample or a distribution is its **kurtosis** which describes how flat a distribution is on its top. The flatness is defined – what else could we possibly do – relative to the "flatness" of a normal distribution. A distribution with positive kurtosis looks like the Matterhorn, a distribution with negative kurtosis looks like a mesa in New Mexico. The first is called *leptocurtic*, the latter *platykurtic*, "and, you will no doubt be pleased to hear, an in-between distribution is termed *mesokurtic*" (*Press et al.*, 1989).

Definition 7.9. Fisher's definition of **kurtosis** is given by

$$\mathrm{kurtosis}(x_1, x_2, \ldots, x_n) \doteq \left\{ \frac{1}{n} \sum_{i=1}^{n} \left[\frac{x_i - \bar{x}}{\sigma} \right]^4 \right\} - 3, \tag{7.6}$$

where $\sigma = \sigma(x_1, x_2, \ldots, x_n)$ is again the sample standard deviation and the -3 term makes the value vanish for a normal distribution. There is also Pearson's definition of kurtosis which does not subtract the 3. So be careful when considering the kurtosis of your data.

The standard deviation of eq. 7.6 as an estimator of the kurtosis of an underlying normal distribution is $\sqrt{24/n}$, but again you should not trust a kurtosis unless it is much larger than that.

`scipy` also provides a routine to determine a sample's kurtosis

```
In:  from scipy.stats import kurtosis
In:  from numpy.random import randn
In:  a = randn(25)
In:  kurtosis(a)
Out: -0.17020793396691625
```

There are bells and whistles to `scipy` 's implementation of the kurtosis (and skew) routines, look up the documentation! As for skewness, `scipy` also provides a test for kurtosis,

```
In:  from scipy.stats import kurtosistest
In:  from numpy.random import randn
In:  a = randn(25)
In:  kurtosistest(a)
```

```
Out: KurtosistestResult(statistic=0.32232765823657772, \\
     pvalue=0.74720448207756052),
```

where the same comments apply as for the skewness test.

7.4 Empirical Covariance and Empirical Correlation Coefficient

The empirical covariance and empirical correlation coefficient are defined for n pairs of observations, $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.

Definition 7.10. One defines the **empirical covariance** as

$$\mathrm{Cov}_{\mathrm{emp}}\,[x, y] \doteq \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}). \tag{7.7}$$

Definition 7.11. The **empirical correlation coefficient** r is then defined as

$$r \doteq \frac{\mathrm{Cov}_{\mathrm{emp}}\,[x, y]}{s_x s_y}, \qquad r \in [-1, +1]. \tag{7.8}$$

It is also called **Pearson's correlation coefficient**.

Beware that it is difficult to decide whether a correlation $r > 0$ is statistically significant based on r alone. It has no knowledge of the distributions underlying x and y. One often hears that a correlation is significant if it is larger than, say, $1/2$ or even $2/3$. While this may indicate a correlation, it does not necessitate a correlation! If x and y are uncorrelated and the tails of their distributions fall off "fast enough" and the number of observations is sufficiently large, say, $n > 20$, then one can say that r is distributed approximately normally with a mean of zero and a standard deviation of $\sigma = 1/\sqrt{n}$. This allows one to compute the probability that a sample of n uncorrelated pairs will exhibit a correlation coefficient larger than r, namely,

$$p = \mathrm{erfc}\left(\frac{|r|\sqrt{n}}{\sqrt{2}}\right),$$

where $\mathrm{erfc}\,x$ is the complementary error function. We discuss this kind of hypothesis testing in more detail in chapter 8.

`scipy` provides a routine to calculate Pearson's correlation of two data sets (necessarily of the same length).

```
In:  from scipy.stats import pearsonr
In:  from numpy.random import randn
In:  a = randn(25)
In:  b = randn(25)
In:  pearsonr(a,b)
Out: (-0.212, 0.31)
```

The first number in the output is Pearson's correlation coefficient and the second is the probability that two normally distributed random samples would have such a correlation.

Exercise 7.1. Because of this uncertainty in interpreting the significance of a correlation r, other, more robust tests for correlation have been developed. A popular one is Spearman's rank-order correlation. Read up on it in *Press et al.* (1989) or elsewhere!

Example 7.3. Note that correlation is not all there is to data...! *Anscombe* (1973) invented four data sets with near-identical correlation coefficients and linear regression lines. In fact, the data sets also have the same means of x and y and variances of x and y. The statistical properties of these four data sets are summarized in the table below and the data are shown in Figure 7.3 together with their regression lines.

	X_1	Y_1	X_2	Y_2	X_3	Y_3	X_4	Y_4
n	11	11	11	11	11	11	11	11
mean	9.0	7.5	9.0	7.5	9.0	7.5	9.0	7.5
variance	11.0	4.13	11.0	4.13	11.0	4.12	11.0	4.12
r	0.816		0.816		0.816		0.816	
$y(x) = ax + b$	$(a, b) = (0.5, 3.00)$		$(0.5, 3.00)$		$(0.5, 3.00)$		$(0.5, 3.00)$	

Anscombe (1973) made the important point which you shall also not forget: "**Plot your data!**", see Fig. 7.3.

7.5 Maximum-Likelihood Estimation

Sofar, we have simply given definitions of estimators or estimates of certain (important) properties of the population under study. But what makes these estimators "good ones"? In other words, what makes a statistic $u(X_i)$ better than

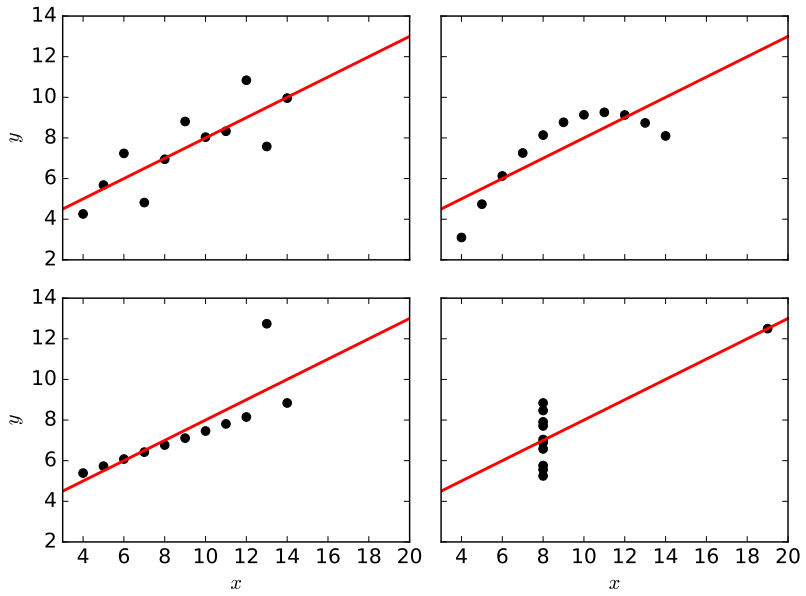


Figure 7.3: When Anscombe's data sets are visualized it is obvious that they are not the same. But their statistical descriptions are near-identical! (*Anscombe*, 1973)

some other statistic $v(X_i)$? Consider the problem at hand. We have a random sample $x_1, x_2, \ldots, x_n$ with an underlying probability density function $p(X)$ which is governed by some paramter, θ. We need to find a good statistic $u(X_i)$ so that $u(x_1, x_2, \ldots, x_n)$ is a good point estimate for θ. As usual, x_i are the observed or measured values of the random variables X_i.

Example 7.4. You plan to take a random sample $X_1, X_2, \ldots, X_n$ of a population. You assume that it is normally distributed (central limit theorem) with mean μ and variance σ^2. You now want to find a "good" estimate of μ and σ^2 using your measured or observed values of X_i, x_i.

So what makes an estimator "good"? Remember that you only have your measurements and a model of the underlying population. One generally accepted criterion is the following. A good estimate of the parameter θ is one that maximizes the probability or *maximizes the likelihood* that you observe the measurements you made. This is the objective of **maximum-likelihood estimation**. Note also how we formulated the problem. A good estimate of the parameter θ maximizes the likelihood of obtaining your data – even if they have uncertainties.

The beauty of this method is that we already have prepared all the necessary ingredients. Suppose that we have a random sample, $x_1, x_2, \ldots, x_n$, for which we

have probability density functions for each X_i, $p(X_i, \theta)$. Then the joint probability density function of $X_1, X_2, \ldots, X_n$ is given as $L(\theta)$

$$L(\theta) \doteq P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = p(x_1, \theta) \cdot p(x_2, \theta) \cdots p(x_n, \theta) = \prod_{i=1}^{n} p(x_i, \theta), \tag{7.9}$$

which treats each random variable as independent because we have drawn independent random samples. All we need to do now to find the maximum-likelihood estimate of θ is to find the value of θ which maximizes the function $L(\theta)$! So we have reduced a complicated and somewhat arbitrary problem (What is a "good" estimate?) to a neat mathematical one that we know how to solve! All we need to do is to minimize the "merit function" (or maximize the likelihood).

Example 7.5. Consider the estimation of the fraction of all students $X_1, X_2, \ldots, X_N$ who are left-handed. We estimate this by observing $n = 100$ students, $x_1, x_2, \ldots, x_{100}$, writing an exam.
This is a classical Bernoulli (or binomial) problem. We have two properties, left- or right-handed, to which we assign values 0 and 1. The X_i are independent and have the unknown parameter, p, and the probability density function for each X_i is

$$\mathcal{P}(x_i, p) = p^{x_i}(1-p)^{1-x_i},$$

for $x_i = 0$ or 1 and $0 \leq p \leq 1$. The likelihood function $L(\mathcal{P}(p)) = L(p)$ is now given by

$$L(\mathcal{P}) = \prod_{i=1}^{n} \mathcal{P}(x_i, p) = p^{x_1}(1-p)^{1-x_1} \cdot p^{x_2}(1-p)^{1-x_2} \cdots p^{x_n}(1-p)^{1-x_n} = p^{\sum x_i}(1-p)^{n-\sum x_i},$$

where $n = 100$ in our example. To maximize the likelihood we need to find the value of p that maximizes $L(p)$, i.e., to take the derivative of $L(p)$ with respect to p and set it equal to zero. Note that we need to take the derivative of a product which is likely to be tedious. We use the conventional trick to take the logarithm of $L(p)$ first and then differentiate. This will still yield the most likely value for p because the logarithm is a continuous and monotonically increasing function. So we now have

$$\ln(L(\mathcal{P})) = \sum_{i=1}^{n} \ln(p^{x_i}) + \sum_{i=1}^{n} (1-p)^{n-x_i} = (\sum_{i=1}^{n} x_i) \ln(p) + (n - \sum_{i=1}^{n} x_i) \ln(1-p).$$

This is easily differentiated with respect to p:

$$\begin{aligned}
\frac{\partial \ln(L(\mathcal{P}))}{\partial p} &= \frac{\sum x_i}{p} - \frac{n - \sum x_i}{1-p} \doteq 0, \\
0 &= p(1-p)\frac{\sum x_i}{p} - p(1-p)\frac{n - \sum x_i}{1-p}, \\
0 &= \sum x_i - p\sum x_i - np + p\sum x_i, \\
0 &= \sum x_i - np \quad \text{or} \\
\hat{p} &= \frac{\sum x_i}{n}. \qquad (7.10)
\end{aligned}$$

So we have found that the *maximum-likelihood estimator* for p is

$$\hat{p} = \frac{\sum X_i}{n},$$

and the *maximum-likelihood estimate* for p is

$$\hat{p} = \frac{\sum x_i}{n},$$

just as we expected.

Note the use of *estimator* with capital letters for the random variables, X_i and *estimate* with small x_is. The first is the statistic acting on the random variable, the second is the estimate based on your measurements. It is only the latter that is measurable. The first is a "concept" and the latter is its realization.

Let us consider a second and somewhat more complicated example.

Example 7.6. Consider a random sample $x_1, x_2, \ldots, x_n$ from a population which is normally distributed with unknown mean μ and variance σ^2. Find the maximum-likelihood estimates for μ and σ^2.
Here we have a probability density function which depends on two parameters, $\theta_1 = \mu$ and $\theta_2 = \sigma^2$,

$$p(x_i; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i - \mu)^2}{2\sigma^2}\right).$$

The likelihood function then is

$$L(\theta_1, \theta_2) = \prod_{i=1}^{n} p(x_i; \theta_1, \theta_2) = \left(\frac{1}{\sqrt{2\pi\theta_2}}\right)^n \exp\left(-\frac{\sum_{i=1}^{n}(x_i - \theta_1)^2}{2\theta_2}\right).$$

The logarithm is

$$\ln\left(L(\theta_1,\theta_2)\right) = -\frac{n}{2}\left(\ln(2\pi)+\ln(\theta_2)\right) - \frac{\sum_{i=1}^{n}(x_i-\theta_1)^2}{2\theta_2}.$$

We first compute the estimate for the mean, $\mu = \theta_1$ by differentiating with respect to θ_1 and requiring this derivative to vanish.

$$0 = 0 - 2\frac{\sum_{i=1}^{n}(x_i-\theta_1)}{2\theta_2}\cdot(-1)$$

Multiplying by $\theta_2 \neq 0$ and dividing by 2 we obtain

$$\sum_{i=1}^{n} x_i - n\theta_1 = 0 \quad \text{and hence} \quad \hat{\theta}_1 = \frac{\sum x_i}{n} = \bar{x},$$

i.e., that the mean or average is the maximum-likelihood estimator for μ.

To obtain the maximum-likelihood estimate of the variance, $\theta_2 = \sigma^2$, of the previous example we proceed in exactly the same way. Take the derivative with respect to θ_2, set it equal to zero and solve for θ_2.

$$\begin{aligned}
\frac{\partial \ln(L(\theta_1,\theta_2))}{\partial\theta_2} &= -\frac{n}{2\theta_2} + \frac{\sum(x_i-\theta_1)^2}{2\theta_2^2}, \\
0 &= -n\theta_2 + \sum(x_i-\theta_1)^2, \\
\hat{\theta}_2 &= \hat{\sigma}^2 = \frac{\sum(x_i-\theta_1)^2}{n},
\end{aligned} \tag{7.11}$$

where we have multiplied by $2\theta_2^2$ in the second line.

Note that the maximum-likelihood equations for normally-distributed errors result in a minimization problem for the sum of squares of the residuals,

$$\frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{2\sigma^2}.$$

The quantity to be minimized is somethimes also called the "merit function" or "objective function". In the case of normally-distributed errors the maximum-likelihood estimator is an ordinary least-squares (OLS) estimator.

7.6 Unbiased Estimators and Estimates

An estimator or estimate should not only maximize the likelihood of the observation, but it should also be *unbiased* in the sense that its expectation value should be the quantity being estimated. More formally,

Definition 7.12. The statistic $u(X_1, X_2, \ldots, X_n)$ is an **unbiased** estimator of the parameter θ if $\mathbb{E}\left[u(X_1, X_2, \ldots, X_n)\right] = \theta$.

Example 7.7. In example 7.5 we showed that $\hat{p} = \frac{\sum x_i}{n}$ was the maximum-likelihood estimate for the fraction of left-handed students. Is it also an unbiased estimate of p?
To show this we compute the expectation value

$$\mathbb{E}\left[\hat{p}\right] = \mathbb{E}\left[\frac{1}{n}\sum X_i\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[X_i\right] = \frac{1}{n}\sum_{i=1}^{n} p = \frac{1}{n}np = p,$$

so $\hat{p}$ is indeed also an unbiased estimator for p.

Let us also consider the maximum-likelihood estimates from example 7.6,

$$\hat{\theta}_1 = \frac{\sum x_i}{n} \quad \text{and} \quad \hat{\theta}_2 = \frac{\sum (x_i - \theta_1)^2}{n},$$

and verify whether they too are unbiased estimates.

Example 7.8. We begin with the easy part, to show that $\hat{\theta}_1 = \frac{\sum x_i}{n}$ is unbiased. We again compute the expectation value,

$$\mathbb{E}\left[\hat{\theta}_1\right] = \mathbb{E}\left[\frac{\sum X_i}{n}\right] = \frac{1}{n}\sum \mathbb{E}\left[X_i\right] = \frac{1}{n}\sum \mu = \frac{1}{n}n\mu = \mu,$$

and so we see that $\hat{\theta}_1 = \frac{\sum x_i}{n}$ is an unbiased and also a maximum-likelihood estimate of the mean.
To show that $\hat{\theta}_2 = \frac{\sum (x_i - \theta_1)^2}{n}$ is an unbiased estimate of σ^2 is a different matter altogether... First we prepare the following

$$\begin{aligned}
\hat{\sigma}^2 &= \frac{\sum (X_i - \theta_1)^2}{n}, \\
&= \frac{1}{n}\sum \left(X_i^2 - 2X_i\theta_1 + \theta_1^2\right), \\
&= \frac{1}{n}\sum X_i^2 - 2\frac{\sum X_i}{n}\theta_1 + \frac{1}{n}\sum \theta_1^2, \\
&= \frac{1}{n}\sum X_i^2 - 2\theta_1\theta_1 + \theta_1^2, \\
\hat{\sigma}^2 &= \frac{1}{n}\sum X_i^2 - \theta_1^2. \qquad (7.12)
\end{aligned}$$

Furthermore, we remind ourselves that the variance $\mathrm{Var}\left[\bar{X}\right] = \mathbb{E}\left[\bar{X}^2\right] - (\mathbb{E}\left[\bar{X}\right])^2$ and thus $\mathbb{E}\left[\bar{X}^2\right] = \mathrm{Var}\left[\bar{X}\right] + (\mathbb{E}\left[\bar{X}\right])^2$ which we use in our computation of the expectation value of the estimator of the variance, σ^2.

$$\begin{aligned}
\mathbb{E}\left[\hat{\sigma}^2\right] &= \mathbb{E}\left[\frac{1}{n}\sum X_i^2 - \theta_1^2\right] = \mathbb{E}\left[\frac{1}{n}\sum X_i^2\right] - \mathbb{E}\left[\theta_1^2\right], \\
&= \frac{1}{n}\mathbb{E}\left[\sum (X_i - \mu + \mu)^2\right] - \mathbb{E}\left[\bar{X}^2\right], \\
&= \frac{1}{n}\sum \left(\mathbb{E}\left[(X_i-\mu)^2\right] - 2\mathbb{E}\left[(X_i-\mu)\mu\right] + \mathbb{E}\left[\mu^2\right]\right) - \mathrm{Var}\left[\bar{X}\right] - (\mathbb{E}\left[\bar{X}\right])^2, \\
&= \frac{1}{n}\sum \left(\mathbb{E}\left[(X_i-\mu)^2\right] - 2\mu\mathbb{E}\left[X_i-\mu\right] + \mathbb{E}\left[\mu^2\right]\right) - \frac{\sigma^2}{n} - \mu^2, \\
&= \frac{1}{n}\left(n\sigma^2 + n\mu^2\right) - \frac{\sigma^2}{n} - \mu^2, \\
&= \sigma^2 + \mu^2 - \frac{\sigma^2}{n} - \mu^2, \\
\mathbb{E}\left[\hat{\sigma}^2\right] &= \sigma^2 - \frac{\sigma^2}{n} = \frac{n-1}{n}\sigma^2 \neq \sigma^2. \qquad (7.13)
\end{aligned}$$

In other words, the maximum-likelihood estimator of σ^2 is not an unbiased estimator!

So what shall we do to find an unbiased estimator of the variance? The easiest way out is to notice that $\mathbb{E}\left[\hat{\sigma}^2\right]$ only differs from $\hat{\sigma}^2$ by a factor $(n-1)/n$. Remember the definition of the variance

$$\sigma^2 = \frac{1}{n}\sum (x_i - \mu)^2.$$

We can easily account for the correction factor by defining a new variance,

$$s^2 \doteq \frac{1}{n-1}\sum (x_i - \bar{x})^2.$$

Obviously, $s^2 = \frac{n}{n-1}\sigma^2$, and so s^2 is now an unbiased estimate for the sample variance.

Exercise 7.2. Verify that this is, indeed, true.

The correction $\sigma^2 \longrightarrow s^2$ or rather to use $n-1$ instead of n in the denominator is called *Bessel's correction*. It corrects the bias in the estimation of the population variance, and *partially* corrects it for the standard variation. An intuitive explanation for the reason to use $n-1$ instead of n is the following. Consider

a population with unknown mean and variance. You now take n independent samples and compute the sample mean and the residuals:

$$(x_1 - \bar{x}), (x_2 - \bar{x}), \ldots, (x_n - \bar{x})$$

You have n residuals, but how many *independent* residuals do you have here? Consider taking the sum of the residuals:

$$\sum_{i=1}^{n}(x_i - \bar{x}) = \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} = n\bar{x} - n\bar{x} = 0.$$

The residuals all make use of one common sample mean to which they all contribute. Sloppily speaking, the sample mean has "used up one degree of freedom". So the "correct" number of degrees of freedom to use when computing the sample variance is $n - 1$. Note that you need to make Bessel's correction *only if you are estimating both the population mean and variance from the same sample.*

I'd like to cut this discussion short now and quote *Press et al.* (1989) who write: "`We might also comment that if the difference between` N `and` $N-1$ `ever matters to you, then you are probably up to no good anyway - e.g., trying to substantiate a questionable hypothesis with marginal data.`"

7.7 Robust Estimation

Usual descriptive statistics make an important assumption, namely that we know the model which underlies the data we are considering. In most cases, this model is a normal distribution, i.e., we implicitly assume that the errors are normally distributed. Real-life experience shows that i) we rarely know the underlying model as well as we should, and ii) there are outlier data points in the data more often than expected. These two points are often connected. Robust statistics attempt to provide methods which can deal with outliers and deviations from the models and still provide meaningful results/estimates.

The problem with usual estimators is that they often perform poorly when outliers are present. We already saw this in Fig. 7.2 which compares the performance of the median and the mean in the presence of outliers. There I mixed two normal distributions with the same mean but different variances with the wider distribution accounting for 10% of the narrow distribution (this is the black curve in Fig. 7.2). This mimicked the presence of occasional outliers in the data.

Figure 7.4 shows some classical data from Newcomb's measurements of the speed of light. He measured the time it took light to pass from his laboratory at the United States Naval Observatory to a mirror attached to the Washington Monument and back again to his laboratory. You can see that the results are

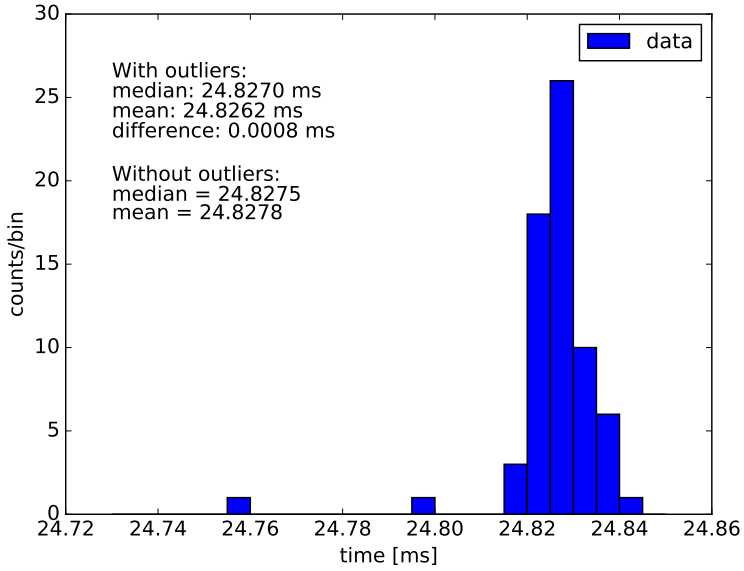


Figure 7.4: Histogram of the light travel-time measurements taken by Newcomb.

quite precise (fractions of milliseconds) but that there are also two outliers in the data. Obviously, these will have consequences for the final result, as is shown in the numbers for the mean and the median in Fig. 7.4.

There are two important concepts in robust estimation, the breakdown point and the influence function.

7.7.1 The Breakdown Point

Definition 7.13. The breakdown point of an estimator (the mean or the median, for example) is the fraction of outliers or incorrect measurements which it can tolerate before it yields incorrect results.

Example 7.9. The mean has a breakdown point of 0%. Consider $x_1, x_2, \ldots, x_n$ observations. Then the mean, computed as $\bar{x} = (1/n)\sum x_i$ can be arbitrarily wrong by having only one single corrupted observation, x_i.

The larger the breakdown point of an estimator, the better. Obviously, the breakdown point cannot exceed 50%. If more than 50% of the observations are corrupted, it is impossible to distinguish which are the "real" and which are the "wrong" measurements. The median has a breakdown point of 50%. The X%-trimmed mean has a breakdown point of X%. It is computed by removing X% of the data from both ends of the data and then computing the mean.

Example 7.10. Consider again the speed-of-light measurements. Removing the two outliers results in a much-improved mean which is comparable with the median.

7.7.2 The Influence Function

In section 7.5 we saw that assuming an underlying normal distribution, minimizing the sum of the square of the differences, $\sum(x_i - \mu)^2$, is a maximum-likelihood estimator of the mean, μ. This is often called the "Ordinary Least Squares (OLS)" method (see page 81). The difficulty of OLS in the presence of outliers is that their weight increases quadratically with their "distance" to the empirical mean, $\bar{x}$, which is called residual (see page 84). This means that outliers with arbitrarily large residuals can have an arbitrarily large influence on the resulting estimate.

One way out of this is the use of so-called M-estimators, which limit the influence of outliers by replacing the square of the residuals by a merit function which increases less rapidly with the residual. This new "merit function" or "objective function" is then called an "influence function". They are in a sense a generalization of the concept of maximum-likelihood estimators, the difference being that they make no pretence at having any relation to the probability density function presumably underlying the data. While maximum-likelihood estimators try to maximize $\prod_i p(x_i)$ or, equivalently, minimize $-\sum_i \log p(x_i)$, M-estimates generalize this concept to the minimization of $\sum_i \rho(x_i)$ where ρ is some suitably chosen function. This minimization of $\sum_i \rho(x_i)$ corresponds to solving $\sum_i \Psi(x_i) = 0$ where $\Psi(x_i) = \mathrm{d}\rho(x)/\mathrm{d}x$ if $\Psi(x)$ exists.

There are many possibilities of M-estimators, four examples are shown in eqs. 7.14 to 7.17.

$$\rho(x) = x^2 \qquad \Psi(x) = 2x, \tag{7.14}$$

$$\rho(x) = |x| \qquad \Psi(x) = x/|x|, \tag{7.15}$$

$$\rho(x) = \begin{cases} x^2 & \text{for} |x| < c \\ |x| & \text{for} |x| \geq c \end{cases} \qquad \Psi(x) = \begin{cases} x/|x| & \text{for} |x| < c \\ 2cx/|x| & \text{for} |x| \geq c \end{cases} \tag{7.16}$$

$$\rho(x) = \begin{cases} \frac{x^6}{6c^4} - \frac{r^4}{2c^2} + \frac{x^2}{2} & \text{for} |x| < c \\ \frac{c^2}{6} & \text{for} |x| \geq c \end{cases} \qquad \Psi(x) = \begin{cases} x\left(1 - \frac{x^2}{c^2}\right)^2 & \text{for} |x| < c \\ 0 & \text{for} |x| \geq c \end{cases} \tag{7.17}$$

Eq. 7.14 is nothing else than OLS and only serves as an illustration. Eq. 7.15 is an M-estimate which minimizes the absolute deviation, $|x|$, instead of the square, x^2. Eq. 7.16 is called Windsoring at a threshold c. For residuals which are less than the threshold, Windsoring ensures an objective function which increases like a least-squares function, for larger residuals, it only increases linearly. Eq. 7.17 is Tukey's bi-weight. It is defined by its $\Psi(x)$ function and its $\rho(x)$ function must be

found by integration. It increases rapidly for residuals $|x| < c$ and is constant to $|x| \geq c$.

Because Tukey's biweight is defined by its $\Psi(x)$, we need to find ist $\rho(x)$ by the integral given in eq. 7.18.

$$\int \Psi(x)\mathrm{d}x = \int x\left(1-\frac{x^2}{c^2}\right)^2 \mathrm{d}x. \tag{7.18}$$

This is easily found by making the substitution

$$u = 1-\frac{x^2}{c^2} \quad \Longleftrightarrow \quad \mathrm{d}u = -\frac{2x}{c^2}\mathrm{d}x,$$

and inserting

$$x = c\sqrt{1-u}$$

into eq. 7.18 to obtain

$$\begin{aligned}\int x\left(1-\frac{x^2}{c^2}\right)^2 \mathrm{d}x &= -\frac{c^2}{2}\int u^2\mathrm{d}u = -\frac{c^2}{2}\frac{u^3}{3}, \\ &= -\frac{c^2}{6}\left(1-\frac{x^2}{c^2}\right)^3, \\ &= +\frac{x^6}{6c^4}-\frac{x^4}{2c^2}+\frac{x^2}{2}-\frac{c^2}{6}, \end{aligned} \tag{7.19}$$

where the constant $c^2/6$ is unimportant because it is constant and it is the derivative which counts for minimization.

The influence functions are shown in Fig. 7.5.

7.8 Quantiles

The median is the simplest example of a quantile, it is the 2-quantile. 50% of all measurements lie below it, 50% above. This concept can be generalized to different quantiles: tertiles (values for $33.\overline{3}$% bins), quartiles (values for 25% bins), etc.

Definition 7.14. Quantiles are cut points that divide the distribution into intervals with the same number or fraction of measurements. The k-th q-quantile is the value along the x-axis, i.e., the data value for which the cumulative distribution function crosses k/q. In other words, x is the k-th q-quantile if $\mathrm{cdf}(x) = k/q$.

Thus there are three 4-quantiles or cut points for the four quartiles of a normal distribution, as is shown in Fig. 7.6.

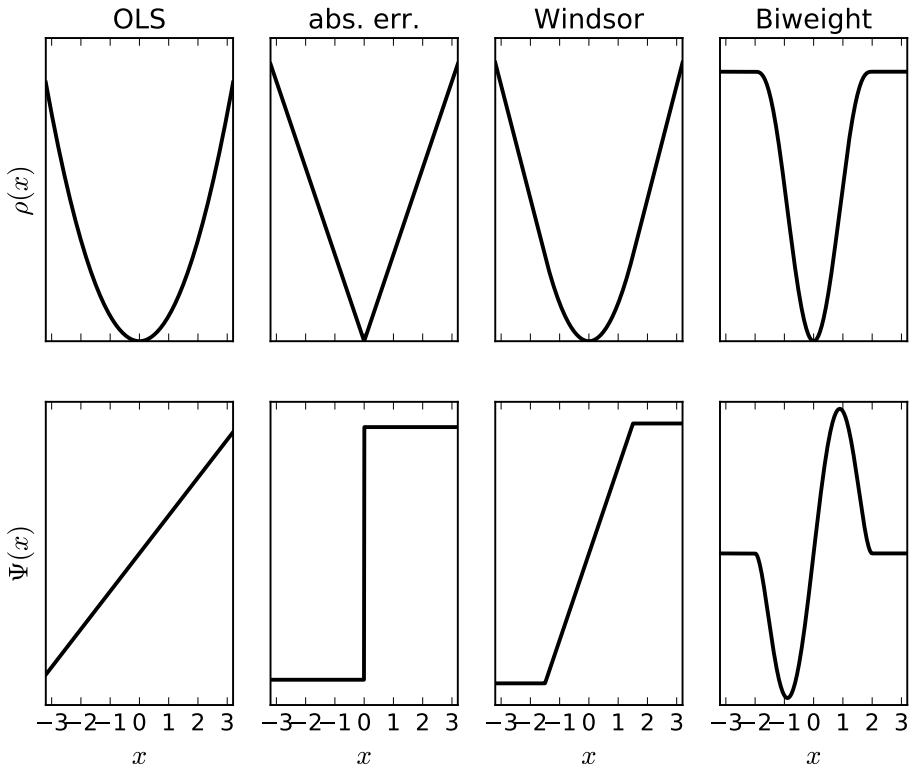


Figure 7.5: Summary plot of various influence functions. From left to right: ordinary least squares (OLS), absolute deviation, Windsoring, and Tukey's biweight. The top row shoes the objective function, the bottom row its derivative.

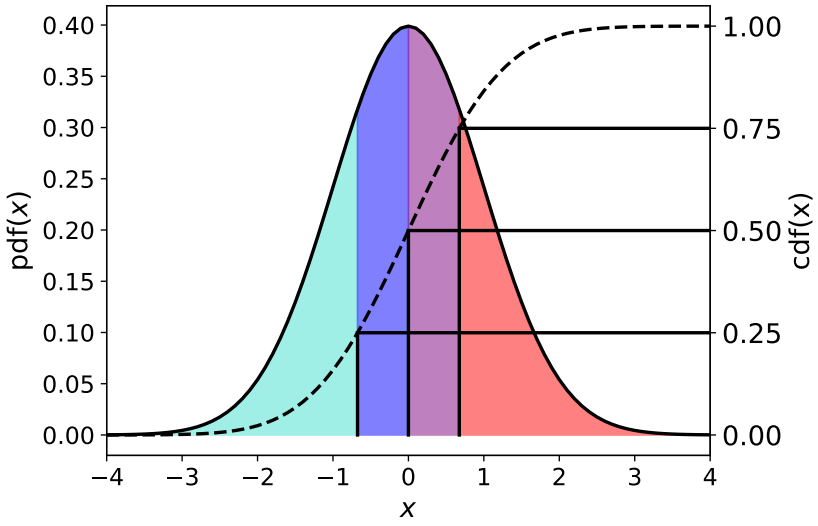


Figure 7.6: Illustration of the concept of quantiles. The shaded areas under the probability density function (pdf) all have the same area, i.e., they contain the same fraction of all measurements. This plot illustrates quartiles of a normal distribution. The quartiles are −0.674, 0, and +0.674.

7.9 Binning Your Data – To Bin or not to Bin

Before you believe any statistical result about your data – **plot it!** But how? Fig. 7.7 shows a time series of the bulk speed and thermal speed of α-particles in the solar wind as measured by the Solar Wind Ion composition Spectrometer (SWICS) (*Gloeckler et al.*, 1998) instrument on the Advanced Composition Explorer (ACE) (*Stone et al.*, 1998).

Let us see what we can do to describe these data statistically. The easiest things to do are to determine the mean and variance of these solar wind properties. You can download the data from the ACE Science Center `http://www.srl.caltech.edu/ACE/ASC/level2/`. I have saved them as a file called ACE_SWICS2_Data-for-2016.txt.

```
In:  from scipy.stats import describe
In:  from numpy import loadtxt
In:  doy, vHe2, vthHe2, vHe2_err, vthHe2_err, qf_He =
     loadtxt('ACE_SWICS2_Data-for-2016.txt',
     skiprows=31,unpack=True)
In:  m = (qf_He == 0)*(vHe2 > 0.) #quality flag.
     #Only use values with qf\_He == 0, i.e.,
     #within SWICS sensitivity range.
```

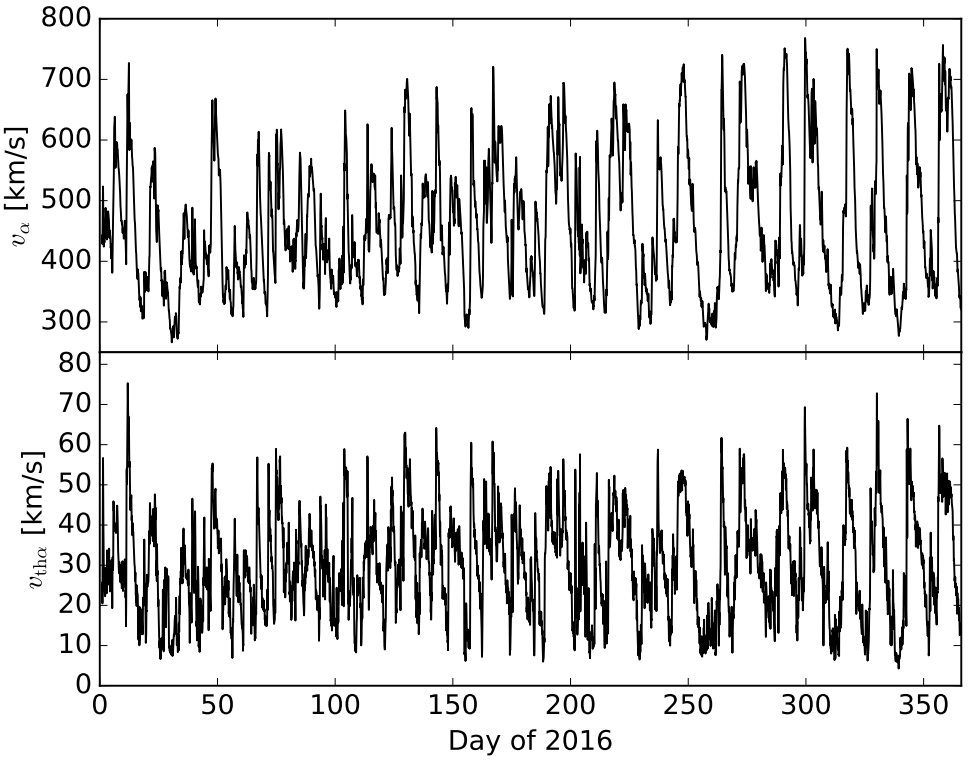


Figure 7.7: Bulk speed (top panel) and thermal speed (bottom panel) of α-particles in the solar wind vs. time as measured by the Solar Wind Ion Composition Spectrometer (SWICS, *Gloeckler et al.* (1998)) in 2016. The data are from the ACE Science Center, see text for url.

```
In:  describe(vHe2)
Out: DescribeResult(nobs=4377, minmax=(266.37, 768.34),
     mean=461.7733, variance=12487.177, skewness=0.588,
     kurtosis=-0.537)
In:  from scipy.stats import skewtest
In:  skewtest(vHe2[m])
Out: SkewtestResult(statistic=14.794, pvalue=1.61e-49)
In:  from scipy.stats import kurtosistest
In:  kurtosistest(vHe2[m])
Out: KurtosistestResult(statistic=-10.065, pvalue=7.88e-24)
```

The bulk α-particle solar wind speed varies around a mean, $\bar{v}_{\mathrm{He2}} = 461.77$ km/s with a variance $\sigma^2 = 12487.177$ (km/s)^2. The distribution is skewed towards higher solar wind speeds and is platykurtic. The probabilities that a random sample (drawn from a normal distribution) would show such skewness and kurtosis is extremely small. But is this a sensible thing to do? Does it really make sense to compute the average solar wind speed? Are variance, skewness and kurtosis sensible ways of describing the distribution of the solar wind?

One obvious thing to do would be to plot thermal speed vs. bulk speed and look for a possible correlation. This is easily done and we observe in Fig. 7.8 that thermal speed increases with bulk speed. We can test for correlation

```
In:  from scipy.stats import pearsonr
In:  pearsonr(vHe2[m],vthHe2[m])
Out: (0.895, 0.0)
```

which tells us that indeed, the two quantities are correlated with a correlation coefficient (eq. 7.8) $r = 0.895$. The probability for two random variables drawn from two normal distributions to show such a correlation is zero (the second number in the output).

Exercise 7.3. Download data from the ACE Science Center and perform these statistical tests and plot the data accordingly.

Inspection of your plot (fig. 7.8) which shows thermal speed vs. bulk speed will show that $v_{\mathrm{th}\alpha}$ is not a linear function of v_α. Moreover, it shows that there appears to be a problem with the determination of solar wind speeds and thermal speeds which results in a stripe-like pattern extending from low to high thermal speeds. This is – once again – a warning: **Always plot your data!**

We have sofar nicely avoided the question posed at the beginning of this section, i.e., how to plot the data so we can say whether the nice statistical analysis results from `scipy` are reliable. This then, brings us to one of the murkiest waters in data

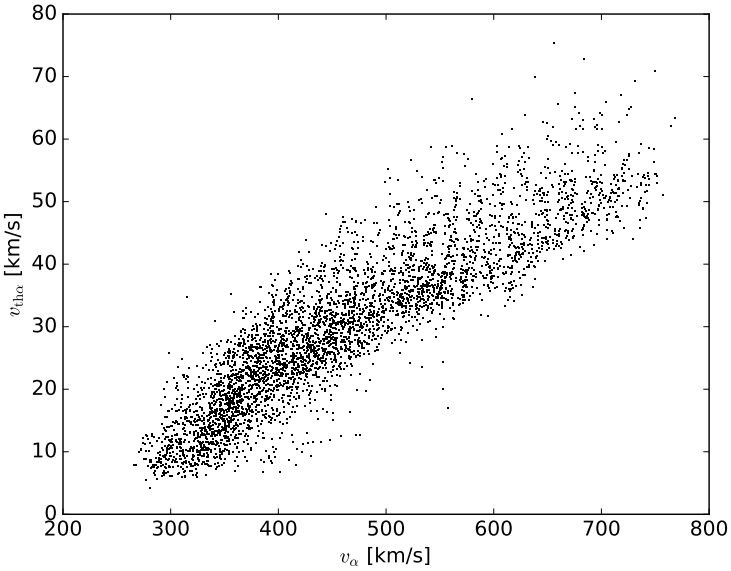


Figure 7.8: Thermal speed vs. solar wind speed (of α-particles). Data are from the ACE Science Center as described in the text.

analysis, the binning of data. This is done very often, and very often it hides more than it reveals...

In Fig. 7.4 I binned the Newcomb's measurements of the travel time of light between his lab and the Washington monument and back again into 25 bins. The result looks OK. But note that I introduced a bias here. I could have chosen a finer binning or a coarser binning and the plot may not have looked the same. Let's do this with the solar wind data of the previous example. Figure 7.9 shows histograms of the bulk speed of α-particles using varying bin widths but always the same minimum and maximum of the bins. The upper left plot was made using `matplotlib`'s `hist` plot routine, for the others I first used `numpy` 's histogram routine with self-defined bins and then plotted the results. The top right histogram has 10 bins, the lower left has 20, and the lower right 40.

Figure 7.10 shows the influence of how one bins. The top left panel shows the results for 40 bins stretching from the minimum to the maximum of the α-particle bulk speeds, i.e., from 266.37 km/s to 768.34 km/s (see results in the listing shown on page 91). The top right panel shows the histogram with 40 bins stretching from $x_{\mathrm{min}} - 5$ km/s to $x_{\mathrm{max}} + 5$ km/s. The bottom left panel shows the histogram with 40 bins stretching from $x_{\mathrm{min}} - 5$ km/s to $x_{\mathrm{max}} - 5$ km/s, and the bottom right one stretches from $x_{\mathrm{min}} + 5$ km/s to $x_{\mathrm{max}} + 5$ km/s. Note that the bin width with 40 bins is about 12.5 km/s, considerably wider than the 5 km/s shifts which I

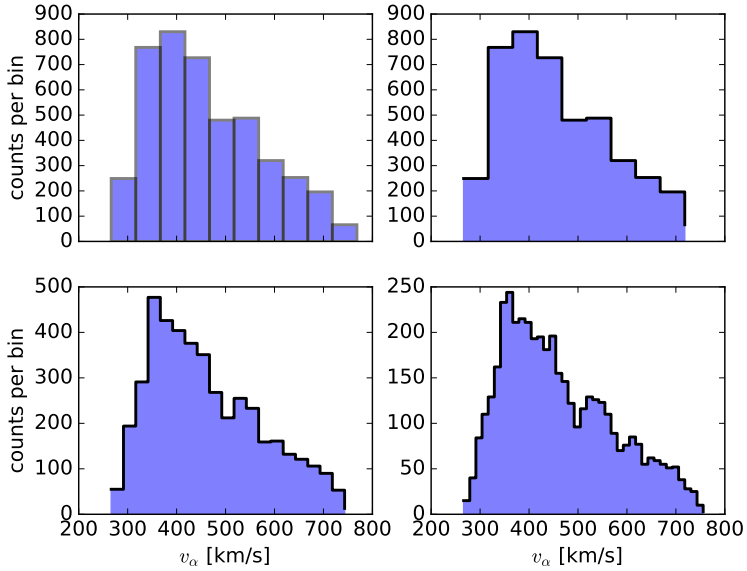


Figure 7.9: Histograms of solar wind α-particle bulk speeds.

made. Nevertheless, the histograms look different, and different "features" appear for different binning[1]. There is also an other difference between the histograms in Fig. 7.9 and 7.10, and that is the labeling of the y-axis. It has relatively meaningful physical units instead of "counts per bin".

The fact that different binning of the same data results in different histograms is by no means rare. It is also the reason why I called binning of data one of the murkiest waters in data analysis. Not every feature which is visible when you bin your data is real. You can also hide features in your data by appropriate binning and choice of the y-axis, but that is something I'm not going to teach you here. Binning is useful if you don't have a model for the underlying distribution. It is a powerful tool to explore data, but it can also be deceptive, so use it with caution. Don't forget that you are throwing away useful information by binning. If you know the underlying distribution, you are better off to keep the full information and fit the distribution to the data, a technique which is presented in chapter 9.

When binning your data – *always* – try several different binnings and *only* trust features which are visible with *all* binnings!

[1]These are probably caused by the "stripes" in solar wind speed seen in fig. 7.8.

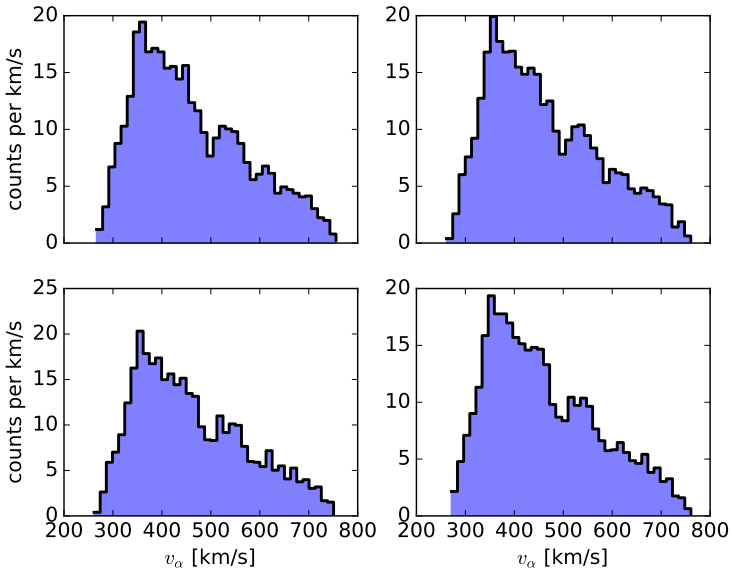


Figure 7.10: Histograms of solar wind α-particle bulk speeds using the same number of different bins. See text for discussion.

Chapter 8

Hypothesis Testing

8.1 Simple Hypothesis Testing

We are often faced with a situation in which we need to decide whether something is "true" or "false" and need to decide this on the basis of statistical data, of a sample of the underlying population. In many cases, such decisions are possible if we allow for two kinds of errors:

1. Type I Error: We erroneously reject a true hypothesis
2. Type II Error: We erroneously accept a false hypothesis

It turns out that one can even give a probability for the two errors - *if we understand the problem!*

The procedure is always the same: We postulate a **null hypothesis**, $\mathbb{H}_0$, and use a statistical test to reject or accept it. Often, $\mathbb{H}_0$ is tested against an **alternative hypothesis**, $\mathbb{H}_1$. As a general rule, one also defines a **significance level**, α, *before one performs the statistical test.* One then performs the statistical test and compares the results with a test statistic. If the probability of the result is less than α, then one rejects the null hypothesis, and says that the null hypothesis is rejected at a significance level of α. If the probability is larger, the we cannot reject the null hypothesis and we say that $\mathbb{H}_0$ is accepted at significance level α or that we cannot reject $\mathbb{H}_0$ at significance level α. More about this after a few first examples.

Example 8.1. Let us again consider our good old die which in the past weeks has been showing some strange behavior. Let us define the null hypothesis that the probability of throwing a "6" is $p(6) = 1/6$, in other words, $\mathbb{H}_0 : p(6) = 1/6$. We have a faint suspicion that a "6" may be more probable, so we define the alternative hypothesis, $\mathbb{H}_1 : p(6) > 1/6$. We also define the significance level as $\alpha = 5\%$ which

is a typical significance level. It tells us that we have a chance of falsely rejecting $\mathbb{H}_0$ or of falsely accepting $\mathbb{H}_1$ of 5%.
Now assume that we throw the following results:

$$[3, 6, 5, 6, 4, 6, 1, 6, 1, 4]$$

Thus we have four sixes in 10 throws. We can compute the probability of throwing at least four sixes in ten throws,

$$P = \sum_{i=4}^{10} \binom{10}{i} (1/6)^i \ (5/6)^{10-i} = 7.5\%.$$

Because 7.5% is larger than our chosen significance level of $\alpha = 5\%$ we cannot reject the null hypothesis and we report that the data are consistent with a fair die at the 5% significance level. Had we thrown 5 sixes, then the story would be different,

$$P = \sum_{i=5}^{10} \binom{10}{i} (1/6)^i \ (5/6)^{10-i} = 1.8\%$$

which is smaller than our chosen significance level of 5% and we would have reported that we rejected the null hypothesis in favor of the alternative hypothesis at a significance level of 5%. Here the probability of committing a type I error would be 1.8% (if we had done the experiment with a fair die).

Example 8.2. Consider again your coffee machine company. You suspect that your subcontractor is cheating you by supplying you with parts of a lesser quality than contractually agreed. The contract says that there shall be no more than 1 defective part in one hundred parts. You just received a load of 100 parts of which 3 were defective. Is the company cheating you?
You again choose a significance level of 5% and your null hypothesis, $\mathbb{H}_0$ that the company is not cheating you. The alternative hypothesis, $\mathbb{H}_1$, is that the company is cheating you. The probability that three parts out of 100 are defective if the expectation value is 1 out of 100 is

$$P = \sum_{i=3}^{100} \binom{100}{i} \ 0.01^i \ 0.99^{100-i} = 7.9\%,$$

and you cannot reject the null hypothesis and you cannot sue your subcontractor. Two months later you find that you have received a total of 16 defective parts out of 500. Now

$$P = \sum_{i=16}^{500} \binom{500}{i} \ 0.01^i \ 0.99^{500-i} = 0.34\%,$$

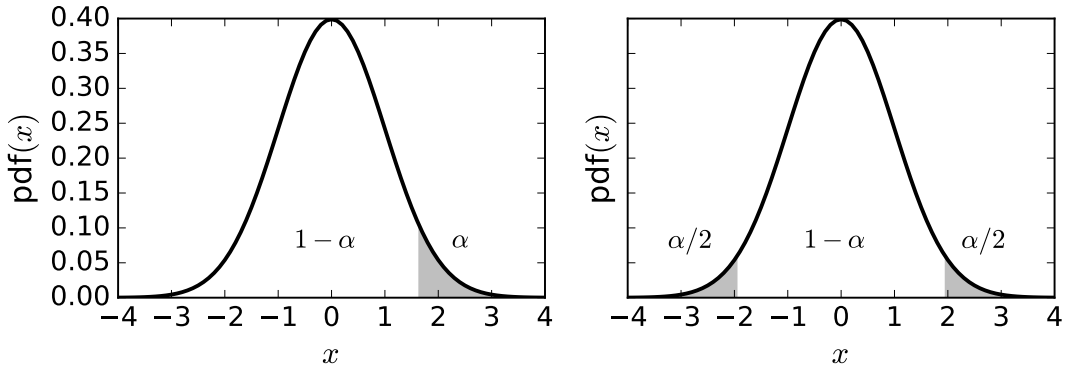


Figure 8.1: Illustration of one-sided (left) and two-sided (right) testing for a significance level $\alpha = 5\%$.

and you go and have a long talk with your lawyers...
Note that we cannot approximate the binomial distribution with a normal distribution because $np < 5$.

Example 8.3. Let us consider another distribution but again with your favorite company. You normally sell 4 coffee machines per day, but the day after an article about your company in the local newspaper, you sold 9 coffee machines. Is this a significant increase? Again, choose $\alpha = 5\%$. The probability of selling 9 or more coffee machines per day is computed using the Poisson distribution (sec. 5.9),

$$P = \sum_{i=9}^{\infty} \frac{4^i}{i!} \exp(-4) = 2.14\%,$$

and you conclude that this is a significant increase in sales.

The examples we have looked at so far are examples of so-called *one-sided* hypothesis tests. Often, one needs to consider *two-sided* hypothesis tests, for instance because we want to decide whether something is outside specified bounds. This is illustrated in Fig. 8.1. The shaded areas in the left- and right-hand panels cover $\alpha = 5\%$ of the total area under the normal distribution. In the one-sided test (left-hand panel) we test only for measurements or observations which are larger than $1-\alpha$ of the distribution, whereas in the two-sided test (right-hand panel) we test for measurements/observations which lie outside $1-\alpha$ of the distribution, i.e. on both sides of the distribution.

Take a close look at the right-hand panel. You see that the $\alpha/2$ areas on both ends of the distribution start at approximately ± 2, i.e., $2 \times \alpha/2 = 2 \times 2.5\% = 5\%$ of the distribution lies outside about ± 2 of the standard normal distribution $\mathcal{N}(0, 1)$

which is plotted here. This is exactly the rule of thumb discussed on page 49, $\sim 5\%$ of the data lie outside $\pm 2\sigma$ of the distribution. If we went to a smaller significance level, say, $\alpha = 1.25$, then this area would be smaller and the probability of a type I error would diminish.

Example 8.4. An example from your coffee-machine company will help illustrate the need for two-sided tests. You've asked you quality assurance (QA) engineer to investigate some parts (shafts) which have been supplied by your problematic sub-contractor. The specification calls for their diameter to be $d = 2.5 \pm 0.1$ mm so that your coffee machines can operate flawlessly. Out of a delivery of 100 shafts, your QA engineer picks 10 at random and measures their diameter:

$$d = [2.8, 2.55, 2.6, 2.3, 2.4, 2.5, 2.2, 2.8, 2.45, 2.6]$$

He is now faced with two problems:
1) Is the mean consistent with $d_{\text{spec}} = 2.5 \pm 0.1$ mm?
2) Is the distribution around the mean consistent with ± 0.1 mm?
We will consider both problems later on in this chapter. Addressing them requires some more background which we first need to develop.

8.2 Confidence Intervals

Given the discussion of Fig. 8.1 you have already a reasonable feeling for what a confidence interval is. It is an interval which - for a symmetric distribution - is centered on the expectation value and gives you a measure for how likely it is that you will find the mean of your observations inside the confidence level. The problem is that we don't know the true mean of the population and that we don't know the standard deviation (or variance) of the population. All we have is a sample taken from the population and its mean and standard deviation:

$$\overbrace{\mu = \frac{1}{N}\sum_{i=1}^{N} x_i, \quad \sigma = \sqrt{\frac{\sum_{i=1}^{N}(x-\mu)^2}{N}}}^{\text{Full population}} \quad \longrightarrow \overbrace{\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad s = \sqrt{\frac{\sum_{i=1}^{n}(x-\bar{x})^2}{n-1}}}^{\text{Sample}}, \tag{8.1}$$

where N includes every single element of the population, and n the entitites in the sample. Normally, we have $n \ll N$ because we can't know the full population because the test is too expensive or possibly also destructive. It does not make any sense to test the entire population if you break it by doing so...

Assume that we have a sample with *sample* mean $\bar{x}$ and known *population* standard deviation σ. How confident can we be that the sample mean is within a

certain range around the population mean? In other words, we would like to know with a certain confidence level α that

$$P(\bar{x} \in \mu \pm z\frac{\sigma}{\sqrt{n}}) = 1 - \alpha. \tag{8.2}$$

Now, since $|\bar{x} - \mu| = |\mu - \bar{x}|$, this is equivalent to

$$P(\mu \in \bar{x} \pm z\frac{\sigma}{\sqrt{n}}) = 1 - \alpha. \tag{8.3}$$

In other words, if we can find the appropriate value of z we can give a confidence interval in which the mean of the distribution will lie with a probability $1-\alpha$. I'm sure that you remember that the error of the mean is $\sigma/\sqrt{n}$, that is the reason why we use $\pm\sigma/\sqrt{n}$ for the confidence interval of the mean. Now you will also remember that we found that 68% of all normally distributed data points should lie within $\pm\sigma$ of the mean, and 95.4% of all data points within $\pm 2\sigma$. So how large does z need to be that $\bar{x} \pm z\sigma/\sqrt{n}$ contains μ with a probability of $1 - \alpha = 95\%$? This means that

$$\frac{1}{\sqrt{2\pi\sigma^2}} \int_{\bar{x}-z\sigma/\sqrt{n}}^{\bar{x}+z\sigma/\sqrt{n}} \exp\left(-\frac{(x-\bar{x})^2}{2\sigma^2}\right) \mathrm{d}x = 1 - \alpha. \tag{8.4}$$

This value is often tabulated, but you can also find it using `scipy`. In the following example we compute this interval for $\alpha = 5\%$ to find $z = 1.96$:

```
In:  from scipy.stats import norm
In:  norm.interval(0.95)
Out: (-1.96, 1.96)
```

This value z is often called the **z-critical value** and is appropriate *only for normal distributions* because we are making the underlying assumption that the random variable,

$$\frac{\bar{x} - \mu}{\sigma/\sqrt{n}} \sim \mathcal{N}(0, 1),$$

i.e., that this random variable is distributed as a standard normal distribution. I give a few values for the z-critical value in Tab. 8.1.

$1-\alpha$	0.68	0.80	0.90	0.95	0.954	0.99	0.997
z	0.99	1.28	1.64	1.96	2.00	2.58	2.97

Table 8.1: Some typical values of the z-critical value.

Summary 8.1. You can give a confidence interval for the mean of the population to lie within a $1-\alpha$ confidence interval using $\pm z_{1-\alpha}\sigma/\sqrt{n}$ around the sample mean $\bar{x}$, where $z_{1-\alpha}$ is the **z-critical value and** σ is the *known population* standard deviation. If the z-critical value is expressed in numbers of standard deviations, then it is often called a **z-score**.

Now it will hardly come as a surprise to you that the mean, μ, and the variance, σ^2 are two independent random variables. Well, it should, I'm sorry to say. This is only true for a normal distribution. We just saw that we can estimate the probability for the population mean, μ, to lie within a confidence interval around the sample mean, $\bar{x}$. The problem is that we *don't know the population variance*, σ^2! This then brings us to the next distribution, the χ^2-distribution.

8.3 The χ^2-Distribution and χ^2-Tests

Following the discussion leading up to this section, we have a random variable,

$$x - \bar{x},$$

which we assume to be normally distributed around zero. In many cases this will be assured by the central limit theorem, but it is not necessarily so, a counter example are the throws of a die. To estimate the variance of the population we would simply use

$$s^2 = \frac{1}{n}\sum_i^n (x_i - \bar{x},)^2, \quad \text{or} \quad s^2 = \frac{1}{n-1}\sum_i^n (x_i - \bar{x},)^2, \tag{8.5}$$

as we saw in the definition of the empirical variance or sample variance, eq. 7.3. Because s^2 is the square of a normally distributed random variable, *it is no longer normally distributed*, but follows a different distribution, the so-called χ^2-distribution.

Definition 8.1. The χ^2-distribution with k degrees of freedom is the distribution of a sum of the squares of k independent standard normal random variables. In other words, if $X_1, X_2, \ldots, X_k$ are independent, *standard* normal random variables, then the sum of their squares,

$$Q = \sum_{i=1}^{k} X_i^2$$

is distributed according to the χ^2-distribution with k degrees of freedom,

$$Q \sim \chi^2(k).$$

Thus, the χ^2-distribution has one parameter, the number of degrees of freedom, k, which is exactly the number of independent random variables. Note that the χ^2-distribution in a way requires a "normalized" distribution, more precisely, it requires that the variance of the underlying normal distribution, $\sigma^2 = 1$, i.e., it is only a χ^2-distribution for *standard* normally distributed random variables, $X_i \sim \mathcal{N}(0, 1)$.

Let us derive the probability density function, $f(y)$, for the $\chi^2(k = 1)$ distribution, where $y = x^2$. We begin with a standard normal random variable, $X \sim \mathcal{N}(0, 1)$. The probability density function of X is

$$p(x) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-x^2}{2}\right).$$

Then the probability density function, $p(y = x^2)$, is given by

$$\begin{aligned}
p(y) &= \frac{\mathrm{d}}{\mathrm{d}y} P(X^2 \leq y) = \frac{\mathrm{d}}{\mathrm{d}y} P(-\sqrt{y} \leq X \leq \sqrt{y}), \\
&= \frac{\mathrm{d}}{\mathrm{d}y} \frac{1}{\sqrt{2\pi}} \int_{-\sqrt{y}}^{\sqrt{y}} e^{-u^2/2} \mathrm{d}u \quad = \quad \frac{2}{\sqrt{2\pi}} \frac{\mathrm{d}}{\mathrm{d}y} \int_0^{\sqrt{y}} e^{-u^2/2} \mathrm{d}u, \\
&= \frac{2}{\sqrt{2\pi}} e^{-\sqrt{y}^2/2} \frac{\mathrm{d}}{\mathrm{d}y} \sqrt{y} \quad = \quad \frac{2}{\sqrt{2\pi}} e^{-y/2} \frac{1}{2\sqrt{y}}, \\
p(y) &= \frac{e^{-y/2}}{\sqrt{2\pi y}} \quad = \quad \frac{1}{\sqrt{2\pi}} y^{\frac{1}{2}-1} e^{-y/2}, \qquad (8.6)
\end{aligned}$$

where the more complicated last expression is shown in preparation of the more general case with k degrees of freedom. Remember that $\Gamma(1/2) = \sqrt{\pi}$. Then, in preparation of the more general $\chi^2(k)$,

$$p(y) = \frac{1}{\sqrt{2}\Gamma(1/2)} y^{\frac{1}{2}-1} e^{-y/2}. \qquad (8.7)$$

Wikipedia[1] gives an interesting and rather intuitive derivation of the probability distribution function for $\chi^2(k)$. They consider the k samples x_i to be a single point in a k-dimensional (Euclidean) space. The χ^2 distribution of k degrees of freedom will then be given by

$$P(Q)\mathrm{d}Q = \int_\nu \prod_{i=1}^{k} (p(x_i)\mathrm{d}x_i) = \int_\nu \frac{e^{-(x_i^2+x_2^2+\ldots+x_k^2)/2}}{(2\pi)^{k/2}} \mathrm{d}x_1 \mathrm{d}x_2 \ldots \mathrm{d}x_n.$$

[1] `https://en.wikipedia.org/wiki/Proofs_related_to_chi-squared_distribution`

Here ν is the volume of the spherical shell spanned by $Q(x)$ in k space, which in turn is a $k-1$ dimensional entity and for which we have

$$Q = \sum_{i=1}^{k} x_i^2.$$

This is a sphere with radius $R = \sqrt{Q}$ in k-dimensional space. Since this radius is a constant, we can remove it from inside the integral and are left with

$$P(Q)\mathrm{d}Q = \frac{e^{-Q/2}}{(2\pi)^{k/2}} \int_\nu \mathrm{d}x_1 \mathrm{d}x_2 \ldots \mathrm{d}x_k$$

The integral is now nothing else than the area of the sphere in k-dimensional space times its thickness, $\mathrm{d}R$,

$$\mathrm{d}R = \frac{\mathrm{d}Q}{2\sqrt{Q}}.$$

The area of a $k-1$-sphere is given by

$$A = \frac{2R^{k-1}\pi^{k/2}}{\Gamma(k/2)},$$

which you can easily verify for $k = 2$ and $k = 3$ and then use induction to prove it for $k \to k+1$. So

$$P(Q)\mathrm{d}Q = \frac{e^{-Q/2}}{(2\pi)^{k/2}} \cdot \frac{2R^{k-1}\pi^{k/2}}{\Gamma(k/2)} \frac{\mathrm{d}Q}{2\sqrt{Q}} = \frac{1}{2^{k/2}\Gamma(k/2)} Q^{k/2-1} e^{-Q/2} \mathrm{d}Q, \tag{8.8}$$

where $\Gamma(k/2)$ is the Γ-function. Equation 8.8 translates to eq. 8.7 in the case of $k = 1$.

Summary 8.2. The χ^2 distribution with k degrees of freedom is the distribution of the sum of squares of k standard normally distributed random variables, $x_1, x_2, \ldots, x_k$. With $Q = \sum_{i=1}^{k} x_i^2$, the probability density function of the $\chi^2(k)$-distribution is given by

$$P(Q)\mathrm{d}Q = \frac{1}{2^{k/2}\Gamma(k/2)} Q^{k/2-1} e^{-Q/2} \mathrm{d}Q.$$

Plots of some $\chi^2(k)$ distributions are shown in Fig. 8.2. The mean of the χ^2-distribution is $\mu = k$, the variance, $\sigma^2 = 2k$.

To come back to our original question, the variance of the population or of a sample is a χ^2-distributed random variable.

As you can see in Fig. 8.2, the χ^2 distribution is highly skewed. As $k \to \infty$, the χ^2 distribution tends towards a normal distribution. It does so very slowly beacuse it is so highly skewed.

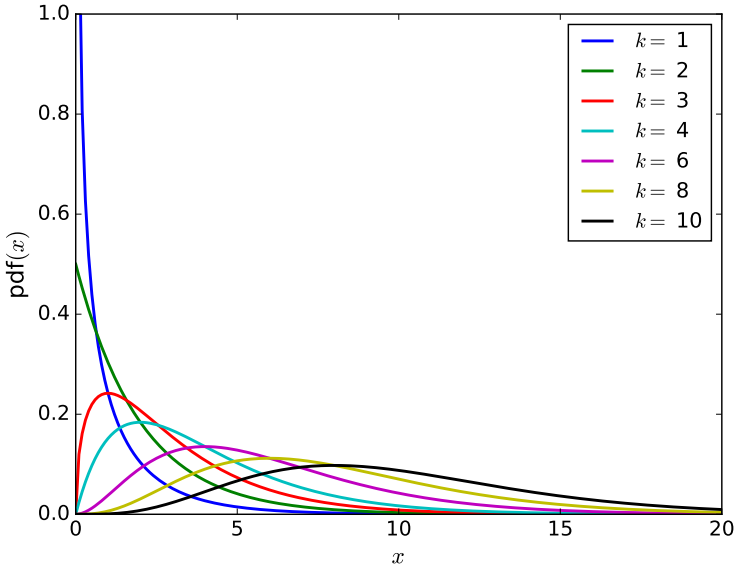


Figure 8.2: Plots of $\chi^2(k)$ for $k = 1, 2, 3, 4, 6, 8, 10$.

Example 8.5. Let us investigate the fairness of our die once again. Our null hypothesis $\mathbb{H}_0$ is that the die is fair, i.e., that all eyes have the same probability. We have thrown it 60 times and observed the following distribution:

eyes:	1	2	3	4	5	6
expected (μ_i):	10	10	10	10	10	10
frequency (x_i):	4	11	10	4	11	20

To decide whether the die is fair, we compare the measurements with their expectation values using a χ^2 test. To do so we compute

$$\begin{aligned}\chi^2 &= \sum_{i=1}^{6} \frac{(x_i - \mu_i)^2}{\mu_i} \\ &= \frac{(4-10)^2 + (11-10)^2 + (10-10)^2 + (4-10)^2 + (11-10)^2 + (20-10)^2}{10} \\ &= 17.4\end{aligned}$$

Let us choose a significance level of $\alpha = 5\%$. The test-statistic, the χ^2 distribution has $f = 6 - 1 = 5$ degrees of freedom. We compute the critical value for a $\chi^2(5)$-distribution at $1 - \alpha$ using

$1-\alpha$	$f=1$	$f=2$	$f=3$	$f=4$	$f=5$	$f=10$
0.68	0.989	2.279	3.506	4.695	5.861	11.499
0.80	1.642	3.219	4.642	5.989	7.289	13.442
0.90	2.706	4.605	6.251	7.779	9.236	15.987
0.95	3.841	5.991	7.815	9.488	11.070	18.307
0.975	5.024	7.378	9.348	11.143	12.833	20.483
0.99	6.635	9.210	11.345	13.277	15.086	23.209

Table 8.2: Critical values for the one-sided χ^2 distribution. Example: The critical value for a χ^2 distribution with $f = 3$ degrees of freedom at a significance level of $1 - \alpha = 0.95$ is 7.815.

```
In:  from scipy.stats import chi2
In:  chi2.ppf(0.95,5)
Out: 11.07
```

We find that the measured χ^2 value exceeds the critical value at a $p = 1-\alpha = 95\%$ significance level, so we can reject our null hypothesis at the 95% confidence level. In fact, Tab. 8.2 shows that the critical levels for 97.5% and 99% are 12.833 and 15.086 which tells us that we can reject the null hypothesis at even higher confidence levels of 97.5% and 99%. We cannot, however, reject it at 99.9% confidence level, because the corresponding critical value is 20.52, as you can easily compute yourself.

How to calculate the degrees of freedom: In the previous example we threw the die 60 times and counted how often we observed each number of eyes. There are 6 eyes on a die, so why did we use $f = 5$? Well, we know that we threw the die 60 times. If we also know how often we had 1, 2, 3, 4, and 5 eyes, then we also know how often we had 6 eyes even without counting the number of times we had 6 eyes. So the number of degrees of freedom is reduced by one, from 6 to 5.

Example 8.6. Let us come back to our coffee machine company[2]. We have recently changed one of the steps in manufacturing in one of the two assembly lines and we wish to find out whether the quality of the machines has improved. We also wonder whether one of the machines in line 2 produces more faulty parts than in line 1. We again carefully collect and analyze all complaints which we have received and categorize them into the following categories:

[2]We should give it a name!

	line 1	line 2	Total
before change	10	13	23
after change	7	11	18
Total	17	24	41

We then go through the following steps:

1. Define the null hypothesis, $\mathbb{H}_0$: There is no difference after the change.
2. Calculate the expectation values: This is slightly more complicated than for the previous example 8.5. We have a total of 41 complaints of which 23 (18) are before (after) and 17 (24) in line 1 (line 2). How do we determine the expectation value? Consider the expected value for line 1 before the change. We have a total of 17 complaints for line 1. Since the null hypothesis is that there is no difference, we would expect the number of complaints to be equally distributed before and after the change. Similarly, if line 1 and line 2 show the same quality, we'd expect the same number in both lines (1 & 2) too. So we multiply 17×23 and divide by the total number of complaints to obtain the expectation value for line1 before the change, $\mathbb{E}\left[\text{line1}, \text{before}\right] = 17 \times 23/41 = 9.54$. Similarly, we obtain the expectation values shown in table 8.3.
3. Next, we calculate the χ^2 of these data and expectation values

$$\begin{aligned}\chi^2 &= \sum_i \frac{(x_i - \mu_i)^2}{\mu_i} \\ &= \frac{(10-9.54)^2}{9.54} + \frac{(13-13.46)^2}{13.46} + \frac{(7-7.46)^2}{7.46} + \frac{(11-10.54)^2}{10.54} \\ &= 0.086\end{aligned}$$

4. Finally, having decided on the conventional $\alpha = 0.05$, we compute the critical value and compare it to the calculated χ^2. The number of degrees of freedom is $f = 1$ and the critical value is 3.84. Here the number of degrees of freedom is calculated as the product of the number of columns minus one and the number of rows minus one, i.e., $k = (n_{\text{col}} - 1)(n_{\text{row}} - 1) = (2-1)(2-1) = 1$.

So we conclude that we cannot reject the null hypothesis, i.e., there is no significant improvement in quality after the change. This result is also valid for our other question, whether the machine in line 2 may produce more faulty parts than the one in line 1.

The Addition theorem for the χ^2 distribution states that if you have $X_1, X_2, \ldots, X_n$ independent χ^2-distributed random variables with $k_1, k_2, \ldots, k_n$ degrees of freedom, then their sum, $Y = \sum_{i_1}^{n} x_i$, is $\chi^2(k_1 + k_2 + \ldots k_n)$ distributed. In other words, $\chi^2(n) = \chi^2(n-1) + \chi^2(1)$.

	line 1	line 2	Total
before change	9.54	13.46	23
after change	7.46	10.54	18
Total	17	24	41

Table 8.3: Expectation values for example 8.6.

8.4 The t-Distribution

In the previous section 8.3 we showed that the variance of a standard normally distributed random variable, X, is distributed as a χ^2 distribution. In other words, the square of a standard normally distributed random variable is no longer distributed as a normal distribution, but as a χ^2 distribution.

Now consider what we are doing when we try to estimate the mean, μ, of a population with variance σ^2 by taking a sample from it. We know that the random variable

$$z = \frac{\bar{x} - \mu}{\sigma/\sqrt{n}}, \quad \text{where} \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

is normally distributed. The problem is, however, that we don't know the variance, σ^2! Well, it is obvious what to do, isn't it? We simply use the empirical variance, s^2, instead of σ^2. We thus have a new random variable

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}, \quad \text{where} \quad s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2. \tag{8.9}$$

Note that the numerator is normally distributed. How is the denominator distributed? This is a question which requires some thought.

Consider again a standard normally distributed random variable, e.g.,

$$\sum_{i=1}^{n}\left(\frac{x_i - \mu}{\sigma}\right) \sim \mathcal{N}(0,1).$$

As we saw a moment ago, its square is $\chi^2(n)$-distributed. Let's write this in a

slightly more complicated way...

$$\begin{aligned} W &= \sum_{i=1}^{n} \left(\frac{x_i - \mu}{\sigma} \right)^2 \sim \chi^2(n), \\ &= \sum_{i=1}^{n} \left(\frac{(x_i - \bar{x}) + (\bar{x} - \mu)}{\sigma} \right)^2, \\ &= \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{\sigma} \right)^2 + \sum_{i=1}^{n} \left(\frac{\bar{x} - \mu}{\sigma} \right)^2 + 2 \left(\frac{\bar{x} - \mu}{\sigma} \right) \sum_{i=1}^{n} (x_i - \bar{x}), \\ &= \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{\sigma} \right)^2 + n \left(\frac{\bar{x} - \mu}{\sigma} \right)^2, \\ &= \frac{(n-1)s^2}{\sigma^2} + n \left(\frac{\bar{x} - \mu}{\sigma} \right)^2. \end{aligned} \tag{8.10}$$

The second term on the right-hand side of eq. 8.10 is

$$\left(\frac{\bar{x} - \mu}{\sigma/\sqrt{n}} \right)^2 \sim \chi^2(1),$$

i.e., is $\chi^2(1)$-distributed. The addition theorem of the χ^2 distribution says that if you have $X_1, X_2, \ldots, X_n$ independent χ^2-distributed random variables with k_1, k_2, $\ldots, k_n$ degrees of freedom, then their sum, $Y = \sum_{i_1}^{n} x_i$, is $\chi^2(k_1 + k_2 + \ldots k_n)$ distributed. In other words, $\chi^2(n) = \chi^2(n-1) + \chi^2(1)$. Because the left-hand side is $\chi^2(n)$ distributed and the second term on the right-hand side is $\chi^2(1)$ distributed, this tells us that the first term on the right-hand side must be $\chi^2(n-1)$ distributed,

$$u = \frac{(n-1)s^2}{\sigma^2} \sim \chi^2(n-1).$$

Now consider again our random variable $t = (\bar{x} - \mu)/(s/\sqrt{n})$.

$$\begin{aligned} t &= \frac{\bar{x} - \mu}{s/\sqrt{n}}, \\ &= \frac{\frac{\bar{x}-\mu}{\sigma/\sqrt{n}}}{\sqrt{\frac{s^2}{\sigma^2}}} = \frac{\frac{\bar{x}-\mu}{\sigma/\sqrt{n}}}{\sqrt{\frac{(n-1)s^2}{\sigma^2(n-1)}}}, \\ t &= \frac{z}{\sqrt{u/(n-1)}}, \end{aligned} \tag{8.11}$$

where $z = (\bar{x} - \mu)/(\sigma/\sqrt{n})$. Thus, our new random variable is the ratio of a standard normally distributed ($\mathcal{N}(0,1)$) random variable and the square root of

a $\chi^2(n-1)$-distributed one. Their distribution functions, $G(z)$ and $\chi^2_{n-1}(u)$ are given by

$$G(z) = \frac{1}{\sqrt{2\pi}} \exp\left(-z^2/2\right) \quad \text{and} \quad \chi^2_{n-1}(u) = \frac{u^{((n-1)/2)-1} e^{-u/2}}{2^{(n-1)/2}\Gamma((n-1)/2)}. \tag{8.12}$$

The probability function, $f_{G\chi^2}(t)$, which describes the distribution of the new random variable, $t = z/\sqrt{u/(n-1)}$ of eq. 8.9 is then

$$f_{G\chi^2_{n-1}}(t) = \int\int \delta\left(t - \frac{z}{\sqrt{u/(n-1)}}\right) G(z)\chi^2_{n-1}(u)\mathrm{d}z\mathrm{d}u, \tag{8.13}$$

where $\delta(t - z/\sqrt{u/(n-1)})$ is the usual Dirac δ-function. Now I fear that the following calculations will be a little painful and I excuse myself for that. First we write the integral in eq. 8.13 in its full glory before we start to simplify.

$$f_{G\chi^2_{n-1}}(t) = \frac{1}{\sqrt{2\pi}} \int\int \delta\left(t - \frac{z}{\sqrt{u/(n-1)}}\right) e^{-z^2/2} \frac{u^{((n-1)/2)-1} e^{-u/2}}{2^{(n-1)/2}\Gamma((n-1)/2)} \mathrm{d}z\mathrm{d}u. \tag{8.14}$$

We now introduce a new variable, $t' = z/\sqrt{u/(n-1)}$ with which $z = \sqrt{u/(n-1)}t'$. Then

$$\frac{\mathrm{d}t'}{\mathrm{d}z} = \sqrt{\frac{n-1}{u}} \quad \Leftrightarrow \mathrm{d}z = \sqrt{\frac{u}{n-1}}\mathrm{d}t',$$

with which eq. 8.14 turns into

$$f_{G\chi^2_{n-1}}(t) = \frac{1}{\sqrt{2\pi(n-1)}} \cdot \frac{1}{2^{(n-1)/2}\Gamma((n-1)/2)} \int \sqrt{u} e^{-\frac{t^2}{2}\frac{u}{n-1}} u^{(n-1)/2-1} e^{-u/2} \mathrm{d}u.$$

Yes, you see that we have gotten rid of one integral and are now left with one more to perform. Let us simplify the integrand

$$f_{G\chi^2}(t) = \frac{1}{\sqrt{2\pi(n-1)}} \cdot \frac{1}{2^{(n-1)/2}\Gamma((n-1)/2)} \int u^{\frac{n-2}{2}} e^{-\frac{u}{2}(1+\frac{t^2}{n-1})} \mathrm{d}u. \tag{8.15}$$

We again introduce a new variable,

$$x \doteq \left(1 + \frac{t^2}{n-1}\right)\frac{u}{2} \quad \text{with which} \quad \frac{\mathrm{d}x}{\mathrm{d}u} = \left(1 + \frac{t^2}{n-1}\right)\frac{1}{2},$$

and

$$u = \frac{2x}{1+\frac{t^2}{n-1}} \quad \text{and} \quad \mathrm{d}u = \frac{2\mathrm{d}x}{1+\frac{t^2}{n-1}}$$

with which eq. 8.15 turns into

$$\begin{aligned} f_{G\chi^2_{n-1}}(t) &= \frac{1}{\sqrt{2\pi(n-1)}} \cdot \frac{1}{2^{(n-1)/2}\Gamma((n-1)/2)} \cdot \frac{2}{1+\frac{t^2}{n-1}} \int \left(\frac{2x}{1+\frac{t^2}{n-1}} \right)^{\frac{n-2}{2}} e^{-x} \mathrm{d}x, \\ &= \frac{\left(1+\frac{t^2}{n-1}\right)^{-(n/2)}}{\sqrt{\pi(n-1)}\Gamma\left(\frac{n-1}{2}\right)} \int x^{\frac{n}{2}-1} e^{-x} \mathrm{d}x. \end{aligned} \tag{8.16}$$

Now the integral

$$\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} \mathrm{d}x,$$

is the Γ function and so we obtain our final result,

$$f_{G\chi^2_{n-1}}(t) = \frac{\left(1+\frac{t^2}{n-1}\right)^{-(n/2)} \Gamma\left(\frac{n}{2}\right)}{\sqrt{\pi(n-1)}\Gamma\left(\frac{n-1}{2}\right)}. \tag{8.17}$$

This distribution is called **Student's t-distribution for $n-1$ degrees of freedom**. It is more convenient to write it as a distribution with n degrees of freedom,

$$f_n(t) = \frac{\left(1+\frac{t^2}{n}\right)^{-(n+1)/2} \Gamma\left(\frac{n+1}{2}\right)}{\sqrt{\pi n}\Gamma\left(\frac{n}{2}\right)}. \tag{8.18}$$

Summary 8.3. Student's t-distribution for n degrees of freedom is given by

$$f_n(t) = \frac{\left(1+\frac{t^2}{n}\right)^{-(n+1)/2} \Gamma\left(\frac{n+1}{2}\right)}{\sqrt{\pi n}\Gamma\left(\frac{n}{2}\right)} \quad \text{where} \quad t = \frac{\bar{x}-\mu}{s/\sqrt{n}}$$

and describes the distribution of the mean of a sample, $\bar{x}$ around the mean of the population, μ, if the variance of the population, σ^2, is not known but sampled from it using the empirical variance, s^2. Figure 8.3 shows the t-distribution.

A graph of Student's t-distributions with various degrees of freedom is shown in Fig. 8.3 where it is also compared with the standard normal distribution (shown as a dashed line). The t-distribution has stronger tails than the normal distribution, thus it allows for larger deviation from "normality".

In order to find the fraction of the t random variable population which lies in a region of the distribution (i.e., to find a certain quantile of the distribution), we

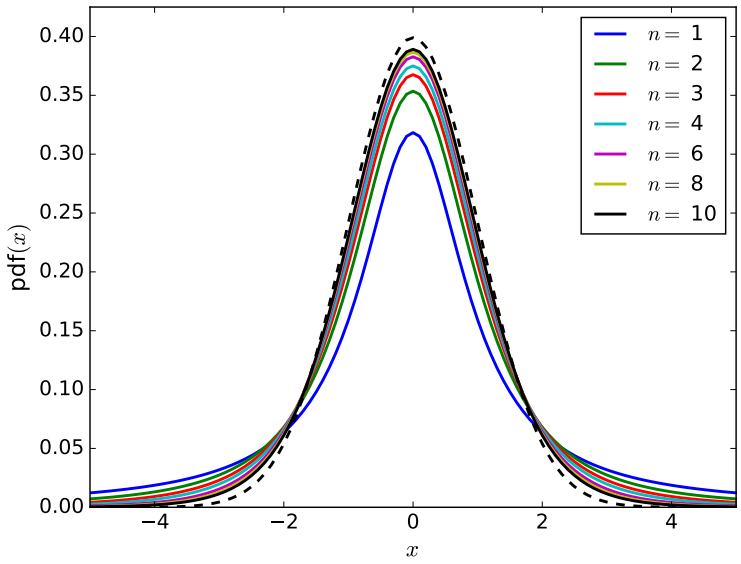


Figure 8.3: Graph of the t-distribution for various degrees of freedom (solid lines) and the standard normal distribution (dashed line).

need to compute the cumulative distribution function. We give it here without proof (Uff! What a relief!).

$$F_n(t) = \int_{-\infty}^{t} f_n(u)\mathrm{d}u = 1 - \frac{1}{2} I_{x(t)}\left(\frac{n}{2}, \frac{1}{2}\right), \quad \text{where} \quad x(t) = \frac{n}{t^2 + n}, \tag{8.19}$$

and I_x is the regularized incomplete beta function,

$$I_x(a, b) = \frac{B(x; a, b)}{B(a, b)},$$

and

$$B(a, b) = \int_0^1 t^{a-1}(1-t)^{b-1}\mathrm{d}t, \quad \text{and} \quad B(x; a, b) = \int_0^x t^{a-1}(1-t)^{b-1}\mathrm{d}t, \tag{8.20}$$

are the beta function and incomplete beta function, respectively. Needless to say, we don't do these calculations on the back of an envelope. Fortunately, `scipy` provides this functionality.

```
In: from scipy.stats import t
In: t.cdf(x, n, loc, scale)
In: t.ppf(x, n, loc, scale)
```

where x and n are the fraction and degrees of freedom of the distribution. `loc` and `scale` allow us to shift and scale the distribution. `ppf` is the inverse of the cumulative distribution function.

Example 8.7. Consider again our coffee machine company for which we have decided to offer a delivery service for high-quality coffee beans to our customers. One of the obvious things which customers do is to weigh the bag of coffee to check whether it contains the 1 kg which we guarantee it should. We weigh 25 out of the 1000 bags and obtain the $\bar{x} = 998$ g for the sample mean weight and $s = 5.3$ g for the standard error of the mean. It seems that we need to fill the bags with a little more than 1 kg to be sure that we won't get too many complaints. But by how much? We want to be sure that no more than 1% of the bags is filled with less than 1 kg of coffee. If we knew the true variance of all the coffee bags, we could use the normal z-critical value, however we don't know σ^2, but only s^2. Because in this situation we need to use the t distribution, we compute the inverse cumulative distribution function or percent point function (ppf) for Student's t for $n = 25 - 1$ degrees of freedom and a confidence level of $\alpha = 1\%$.

```
In:  from scipy.stats import t
In:  t.ppf(0.01,24)
Out: 2.49
```

So we need to multiply the empirical standard deviation by 2.49 and add $2 + 2.49 \cdot 5.3 = 15$ g of coffee to each bag to be sure that we won't have more than $\alpha = 1\%$ underfilled bags. While this looks like a large number to use, you can appreciate that it is needed when you consider the large tails of the t distribution which arise because we don't know the true variance of the population. Had we chosen a larger value for α, $t_{\alpha,n-1}$ would have been smaller.

Note that in example 8.7 we used $t_{\alpha,n-1}$ and not $t_{\alpha/2,n-1}$ as one often sees. This is because we're performing a one-sided hypothesis test! We are only worried about underfilling the coffee bags. If we are considering a two-sided test, we need to use $t_{\alpha/2,n-1}$.

Summary 8.4. Using the definition (eq. 8.9) of the variable $t = (\bar{x} - \mu)/(s/\sqrt{n})$ we can give the $1-\alpha$ confidence interval for the sample mean, $\bar{x}$, and the population mean, μ, to agree within $\pm t_{\alpha/2,n-1}s/\sqrt{n}$. Given the mean, $\bar{x}$, and empirical standard deviation, s, of a sample, $x_1, x_2, \ldots, x_n$, of a population, one reports a $1-\alpha$ confidence interval for the **mean**, μ, of that population as follows:

$$\mu = \bar{x} \pm t_{\alpha/2,n-1}\frac{s}{\sqrt{n}}, \tag{8.21}$$

as can be seen by solving eq. 8.9 for μ. On the other hand, the $1-\alpha$ confidence interval for a new measurement, or the $1-\alpha$ confidence **prediction** interval, is obtained by quadratically adding the standard error of a measurement to the last term in eq. 8.21.

$$\mu = \bar{x} \pm t_{\alpha/2,n-1}s\sqrt{1+\frac{1}{n}}. \tag{8.22}$$

Example 8.8. Consider a sample $x_1, x_2, \ldots, x_n$ of a population with unknown mean μ and variance σ^2. We give the 95% confidence interval for the population mean,

$$\mu = \bar{x} \pm t_{\alpha/2,n-1}\frac{s}{\sqrt{n}},$$

where $\alpha = 0.05$.

8.5 The F-Distribution

This distribution is named after Ronald Fisher, who developed it together with George Snedecor around 1920 for a statistical test whether the *variances* of two distributions are the same. To understand this test we consider two independent normally distributed random variables, X_1 and X_2 with a common variance, $\sigma_1^2 = \sigma_2^2 = \sigma^2$. Now consider the two samples, $x_{1,1}, x_{1,2}, \ldots, x_{1,n_1}$ and $x_{2,1}, x_{2,2}, \ldots, x_{2,n_2}$ which are drawn from the random variables, X_1 and X_2. They have sample means μ_1, μ_2 and empirical variances s_1^2 and s_2^2. Then the quotient

$$q = \frac{s_1^2}{s_2^2}$$

is distributed according to the F-distribution with $n_1 - 1$ and $n_2 - 1$ degrees of freedom. The degrees of freedom are often called numerator and denominator

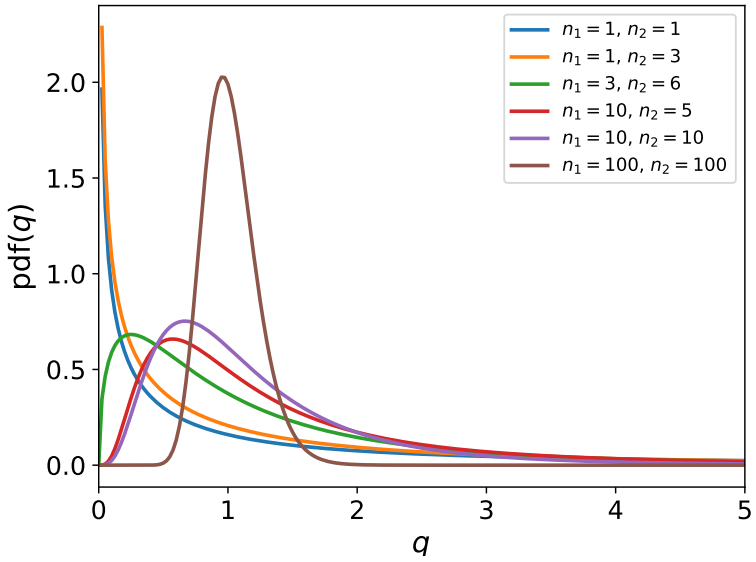


Figure 8.4: Probability density function (pdf) for the F-distribution.

degrees of freedom or `dfn` $= n_1$ and `dfd` $= n_2$. The probability density function of the F-distribution is given without derivation here,

$$p(q) = \frac{n_1^{\frac{n_1}{2}} n_2^{\frac{n_2}{2}}}{qB\left(\frac{n_1}{2}, \frac{n_2}{2}\right)} \cdot \frac{q^{\frac{n_1}{2}}}{(n_2 + n_1 q)^{(n_1+n_2)/2}}, \tag{8.23}$$

where $B(a, b)$ is the beta function given in eq. 8.20. The F-distribution is also provided by `scipy` , plots of its probability density function, $p(q)$, for different n_1 and n_2 are shown in Fig. 8.4.

```
In:  from scipy.stats import f
In:  f.pdf(1.,10,15)
Out: 0.677
In:  f.ppf(0.05,10,15)
Out: 0.351
```

Example 8.9. Consider two independent normally distributed random variables, X_1 and X_2 with unknown means μ_1 and μ_2 and variances σ_1^2 and σ_2^2. We wish to test whether the two random variables have the same variance, i.e., we test for the null hypothesis

$$\mathbb{H}_0 : \sigma_1^2 = \sigma_2^2.$$

We compute the quotient, the ratio

$$q = s_1^2/s_2^2$$

where s_1 and s_2 are the empirical variances of the samples $x_{1,1}, x_{1,2}, \ldots, x_{1,n_1}$ and $x_{2,1}, x_{2,2}, \ldots, x_{2,n_2}$. It is the test statistic for Fisher's F-test. We choose our standard significance level, $\alpha = 5\%$, and the alternative hypothesis

$$\mathbb{H}_1 : \sigma_1 \neq \sigma_2$$

We accept the null hypothesis, $\mathbb{H}_0$, if

$$F(\alpha/2, n_1, n_2) < q < F(1 - \alpha/2, n_1, n_2),$$

where $F(\alpha/2, n_1, n_2)$ is Fisher's F.

Example 8.10. Consider the following two samples,
sample 1 = [2.499, 2.575, 2.758, 3.204, 2.221, 1.714, 3.004, 2.824, 2.473, 2.524]
sample 2 = [3.699, 3.590, 3.366, 2.412, 3.701, 1.255, 1.990]
which have variances $s_1 = 0.173$ and $s_2 = 0.956$, their quotient is $q = 0.1814$. We compute the critical values for the chosen significance level, $\alpha = 5\%$.

```
In:  from scipy.stats import f
In:  n1 = 10; n2 = 7; alpha = 0.05
In:  crit1 = f.ppf(alpha/2,n1-1,n2-1)
In:  crit2 = f.ppf(1.-alpha/2,n1-1,n2-1)
In:  print(crit1, crit2)
Out: 0.2315, 5.5234
```

The quotient lies outside the critical values for the significance level of 5%,

$$q < 0.2315 < 5.5234,$$

so we can say that we *reject* the null hypothesis that $\sigma_1 = \sigma_2$ at the 5% significance level. In fact, I generated the two samples with different variances, $\sigma_1 = 1.0$ and $\sigma_2 = 1.5$, but a common mean of $\mu_1 = \mu_2 = \mu = 3.0$.

Just to check that we did indeed compute a two-sided test for $\alpha = 5\%$, we compute the probability values for the above critical values:

```
In:  print('probability values for critical values are:')
In:  print(f.cdf(crit1,n1-1,n2-1), f.cdf(crit2,n1-1,n2-1))
Out: 0.025 0.975
```

Had we chosen a different significance level, say, $\alpha = 1\%$, we would have found different critical values, namely $c_1 = 0.142$ and $c_2 = 10.39$, and $c_1 < q < c_2$, so we would not have rejected the null hypothesis at this significance level. In this case, we could have determined the average or "pooled" sample variance,

$$s^2_{\text{pooled}} \doteq \frac{s_1^2(n_1 - 1) + s_2^2(n_2 - 1)}{n_1 + n_2 - 2}. \tag{8.24}$$

8.6 Do two Populations Have the Same Mean?

Example 8.11. We have had enough. That machine in line two of our manufacturing hall has indeed been producing faulty parts and we have decided to replace it. As there has been a lot of progress in machine technology, we are thinking of buying a highly capable one and now want to verify that it does indeed produce better parts. To do so, we compare a sample $x_{1,1}, x_{1,2}, \ldots, x_{1,n_1}$ from the corresponding (and older) machine in line 1 which has been producing good parts with a sample $x_{2,1}, x_{2,2}, \ldots, x_{2,n_2}$ from the new machine.

The example just described is fairly general. Note that the sample sizes n_1 and n_2 are not necessarily the same, their means, $\bar{x}_1$ and $\bar{x}_2$, are (probably) different, and their variances, s_1^2 and s_2^2, are also not necessarily the same. The latter can be tested using the F-test which we just discussed a moment ago in sec. 8.5. We may also want to introduce a threshold, $d \doteq \mu_1 - \mu_2$, into the problem, for instance because the new machine is expensive and the parts need to be at least d better to make the new machine worth while. Here μ_i is the mean of population i, not of the corresponding sample! After all, we don't want to be fooled into buying an expensive machine based on a sample which by chance was not a good one.

So how do we decide whether the two means, $\bar{x}_1$ and $\bar{x}_2$, are significantly different? One obvious way of deciding that would be to ask by how many standard deviations the two means differ. In fact, we should not use standard deviations, but the standard error of the two means to compare them. Thus, the test to be used to test for equal means is the **two-sample t-test**. We define our null and alternative hypothesis, as well as the test statistic, etc., in the following table.

$\mathbb{H}_0$:	$\mu_1 = \mu_2$
$\mathbb{H}_1$:	$\mu_1 \neq \mu_2$
$\mathbb{H}_2$:	$\mu_1 < \mu_2$
$\mathbb{H}_3$:	$\mu_1 > \mu_2$
Test statistic:	$T = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n_1 + s_2^2/n_2}}$.
Significance level:	α
Degrees of freedom:	$\nu = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\left(\frac{s_1^2}{n_1}\right)^2 \frac{1}{n_1 - 1} + \left(\frac{s_2^2}{n_2}\right)^2 \frac{1}{n_2 - 1}}$

where the degrees of freedom need to be rounded to the next smaller integer. Often, one assumes the two variances to be equal (as a rule of thumb, when they are within a factor of two from each other). In this case, the test statistic is changed to

$$T = \frac{\bar{x}_1 - \bar{x}_2}{s_p \sqrt{1/n_1 + 1/n_2}}, \quad \text{where} \quad s_p^2 = \frac{s_1^2(n_1 - 1) + s_2^2(n_2 - 1)}{n_1 + n_2 - 2}$$

is the pooled variance (eq. 8.24). In this case

$$\nu = n_1 + n_2 - 2.$$

We define the critical region, i.e., where we reject the null hypothesis if

$$|T| > t_{1-\alpha/2,\nu}, \tag{8.25}$$

where $t_{1-\alpha/2,\nu}$ is the critical value of the t-distribution with ν degrees of freedom.

Example 8.12. We believe that the time the machine in line 2 takes to manufacture a part is longer than machine 1 takes. We have the following samples:

$$\begin{aligned} d_1 &= [29.943, 28.657, 29.478, 32.860, 28.147, 30.291, 29.945, 29.556] \\ d_2 &= [37.113, 35.001, 41.744, 29.464, 42.021, 42.207, 47.827] \end{aligned}$$

We compute the relevant quantities:

quantity	value
$\bar{x}_1$	29.8597
$\bar{x}_2$	39.3394
s_1^2	1.974
s_2^2	35.756
T	-4.0966
$t_{1-\alpha/2,\nu}$	3.5768

where we have computed the critical value for a significance level of $\alpha = 1\%$ because the machine is expensive and we don't want to make a bad decision. Note that the two means clearly look different, the variance of the first sample also looks small and we are inclined to believe that the two samples are indeed different. The problem lies with the second sample and its large variance! Nevertheless, the test statistic is $T = -4.0966$ indicating that $\mu_1 < \mu_2$, and the critical value from the t-distribution is $t_{1-\alpha/2,\nu} = 3.5768$. Thus, we may state that we reject the null hypothesis, $\mathbb{H}_0 : \mu_1 = \mu_2$, at the 1% significance level. But in fact, we can do more than that. We can also test for the alternative hypotheses, $\mathbb{H}_2 : \mu_1 < \mu_2$ and $\mathbb{H}_3 : \mu_1 > \mu_2$. These are now no longer two-sided hypothesis test, but one sided ones (see Fig. 8.1).

alternative hypothesis	rejection region	critical value
$\mathbb{H}_1 : \mu_1 \neq \mu_2$	$\lvert T\rvert > t_{1-\alpha/2,\nu}$	3.5768
$\mathbb{H}_2 : \mu_1 > \mu_2$	$T > t_{1-\alpha,\nu}$	3.0520
$\mathbb{H}_3 : \mu_1 < \mu_2$	$T < t_{\alpha,\nu}$	-3.0520

The test statistic $T = -4.0966 < -3.0520$ and so we may state at the 1% confidence level that $\mu_1 < \mu_2$. Indeed, the samples were generated using normal distributions with means $\mu_1 = 30$ and $\mu_2 = 38$ with standard deviations $\sigma_1 = 1.6$ and $\sigma_2 = 5.4$. The critical value is computed using the following code snippet.

```
In: t.ppf(1.-alpha,n)
```

Example 8.13. *Press et al.* (1989) discuss a final interesting example for the t-test, the case of paired samples. This test is useful if you suspect that part of the variance in both samples is due to a correlation between the samples. They give the example of a committee of of 10 people evaluating two candidates. If some committee members generally give better marks than others, then there will be an additional spread or variance in the data which may mask a difference in the means of the two candidates. *Press et al.* (1989) give the following test statistic,

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s_D}, \quad \text{where} \quad s_D = \sqrt{\frac{\mathrm{Var}\,[S_1] + \mathrm{Var}\,[S_2] - 2\,\mathrm{Cov}_{\mathrm{emp}}\,[S_1, S_2]}{n}},$$

where n is the number of paired data points, S_i are the samples of candidate i, and

$$\mathrm{Cov}_{\mathrm{emp}}\,[S_1, S_2] = \frac{1}{n-1}\sum_{i=1}^{n}(x_{1,i} - \bar{x}_1)(x_{2,i} - \bar{x}_2),$$

is the empirical covariance of the two samples (see eq. 7.3 for the definition). The test statistic, t, is then distributed according to Student's t-distribution and we can use the usual t test.

8.7 ANOVA

ANalysis Of VAriances (ANOVA) is a nifty way of comparing the means of many samples which is conceptually surprisingly simple. Given k random variables, X_i, with expected values $\mu_1, \mu_2, \ldots, \mu_k$, and a *common variance*, σ^2, we take n_i samples of each of these X_i. For each sample, we have a sample mean, $\bar{x}_i$ and empirical variance, s_i^2.

ANOVA tests the null hypothesis $\mathbb{H}_0 : \mu_1 = \mu_2 = \ldots = \mu_k$ and assumes that the individual samples are normally distributed. The basic trick of ANOVA is to estimate the common variance, σ^2 in two independent ways. The first is to add the individual sample variances,

$$s_{\text{tot}}^2 = \frac{1}{n-k} \sum_{i=1}^{k} (n_i - 1) s_i^2. \tag{8.26}$$

Note that this is exactly the way we would add errors quadratically. The second estimate of the variance is to use the individual sample means to compute a common variance,

$$s_{\text{comm}}^2 = \frac{1}{k-1} \sum_{i=1}^{k} n_i (\bar{x}_i - \bar{x})^2, \tag{8.27}$$

where $\bar{x}$ is the mean of all samples. Obviously, s_{tot}^2 will be a good estimate of σ^2 even if the null hypothesis is not true, whereas s_{comm}^2 only estimates σ^2 correctly when $\mathbb{H}_0$ is true. If $\mathbb{H}_0$ is not true, s_{comm}^2 estimates σ^2 too large.

ANOVA uses the test statistic

$$q = \frac{s_{\text{comm}}^2}{s_{\text{tot}}^2} \tag{8.28}$$

which - because both variances are normally distributed - is described by Fisher's F-distribution. Thus, we can use Fisher's test to assess whether our null hypothesis must be rejected or not.

Example 8.14. I created three normally distributed random samples with different lengths, but equal variances, σ_i, and equal expectation values, μ_i. They are given in the following table.

sample	sample values				mean	variance
sample 1	[10.588	7.474	13.268	10.030		
	5.007	10.294]			9.443	6.761
sample 2	[7.984	12.779	9.820	8.592		
	12.558	6.037	7.388	9.150]	9.288	4.944
sample 3	[11.126	7.618	12.251	9.838		
	7.722	7.268	10.080	8.943		
	11.987	8.233]			9.507	3.044

We can now compute $s^2_{\rm tot}$ and $s^2_{\rm comm}$ and the other relevant quantities. The mean of the three sample means is $\bar{x} = (1/3)(\bar{x}_1 + \bar{x}_2 + \bar{x}_3) = 9.413$, $s^2_{\rm tot} = 4.562$, and $s^2_{\rm comm} = 0.109$. Thus, the test statistic $t = s^2_{\rm comm}/s^2_{\rm tot} = 0.02379$. Fisher's distribution is $F(1-\alpha, 2, n_1+n_2+n_3-3) = 3.467$ which is larger than the test statistic, and so we can't reject the null hypothesis. We have used our standard $\alpha = 5\%$ significance level.

If, on the other hand, we add 3.5 to sample 1, we get the following quantities: $\bar{x} = (1/3)(\bar{x}_1 + \bar{x}_2 + \bar{x}_3) = 10.246$, $s^2_{\rm tot} = 4.562$, and $s^2_{\rm comm} = 15.046$. Thus, the test statistic $t = s^2_{\rm comm}/s^2_{\rm tot} = 4.234$. Fisher's distribution is $F(1-\alpha, 2, n_1+n_2+n_3-3) = 3.467$ which is less than the test statistic, and so we reject the null hypothesis.

8.8 Quantile-Quantile Plots

As we saw in Fig. 7.3, is is important to plot the data which one is looking at. To repeat the lesson stated there: **Plot your data!** But how should we do this in the context of hypothesis testing? This is what this section about quantile-quantile plots is about. We will see how to plot data to understand whether our data fulfill the assumptions which we're making.

Fig. 8.5 shows two plots which illustrate a way to check whether our sample data were drawn from a normal distribution. The data in the left-hand panel were, those in the right-hand panel weren't.

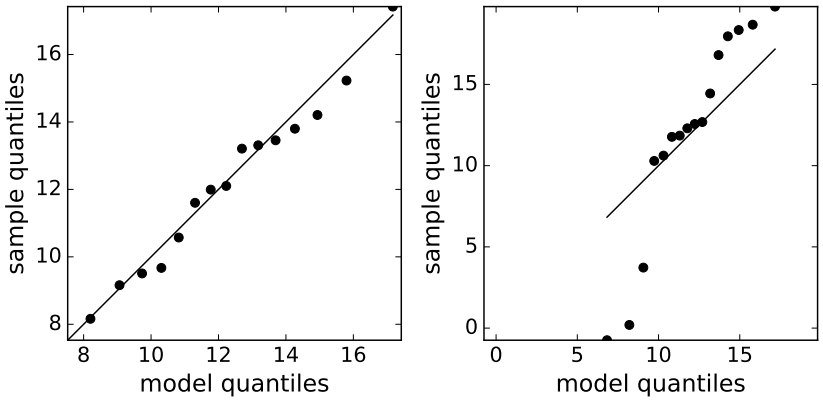


Figure 8.5: "Normality plots" to check whether data are distributed according to a normal distribution. In the left-hand panel, we see data which were generated using a normal distribution, the right-hand panel shows data which was generated as the sum of two normal distributions with different widths.

Such plots are known as **probability plots** and can be generated for any probability function for which quantiles can be calculated. The procedure given

by *Filliben* (1975) and *Chambers et al.* (1983) is as follows:

1. Sort your sample into an ascending order
2. Calculate the uniform order statistic medians, m_i. *Filliben* (1975) gives a simple algorithm

$$m_i = \begin{cases} 1 - m_n & i = 1 \\ (i - .3175)/(n + .365) & i = 2, 3, \ldots, n-1 \\ (1/2)^{1/n} & i = n \end{cases} \tag{8.29}$$

   (In other words, you start by computing $m_n = (1/2)^{1/n}$, the compute $m_1 = 1 - m_n$, and then compute the m_i in between using eq. 8.29.
3. Next compute the order statistic medians for the probability distribution you believe is being sampled. This can be done using the `.ppf` method for the distributions given by `scipy` . For a normal distribution, we'd use `norm.ppf(m,loc=mu,scale=sig)`.
4. Note that we have as many order statistic medians as data (sample) points.
5. Now plot your sorted data vs. the order statistic medians

```
In:  ss = sort(sample)
In:  m = [1.-0.5**(1./n)]
In:  for i in linspace(2,n-1,n-2):
In:      m = append(m,(i-0.3175)/(n+0.365))
In:  m = append(m,0.5**(1./n))
In:  pp = norm.ppf(m,loc=mu,scale=sig)
In:  fig, ax = subplots()
In:  ax.plot(pp,ss,'ko',label='Data')
In:  ax.plot(pp,pp,'k-')
In:  show()
```

In the code snippet above, we used a normal distribution centered around μ with standard deviation σ. We could also have used a Weibull distribution, or any other distribution for which an inverse exists (so you can compute the `.ppf`).

You can also plot two samples against each other to check whether they come from the same distribution. In this case, the probability plot is no longer called a probability plot because you are no longer comparing your sample data with a probability distribution. Such plots are called **quantile-quantile plots** or **Q-Q plots** and compare two (unknown) distributions, i.e., your samples, with each other. The two samples do not need to have the same length.

The recipe to generate a Q-Q plot is the following:

1. Sort both samples into ascending order.

2. Define the uniform order statistic medians, m_i, as defined in eq. 8.29 for both data sets and the number of quantiles which you wich to plot. In other words, you generate three different sets of m_is.

3. Define up to two interpolation functions which allow you to interpolate the two sorted samples for the quantiles which you wish to plot.

4. Now plot the two interpolating functions against each other.

If the two data sets have the same length, you can simply plot them against each other. An example of a q-q plot is shown in Fig. 8.6, the code is shown in the listing below.

```
from pylab import *
from scipy.stats import norm, t
from scipy.interpolate import interp1d
from numpy.random import randn
from numpy import *

n1 = 75
n2 = 58
sig1 = 3.
sig2 = 4.5
mu1 = 12.
mu2 = 25.
cm = 3.
sample1 = mu1 + sig1*randn(n1) - cm*sig2*randn(n1)
sample2 = mu2 + sig1*randn(n2) + cm*sig2*randn(n2)

ss1 = sort(sample1)
ss2 = sort(sample2)


n = 40
m = [1.-0.5**(1./n)]
for i in linspace(2,n-1,n-2):
    m = append(m,(i-0.3175)/(n+0.365))
m = append(m,0.5**(1./n))
```

```
m1 = [1.-0.5**(1./n1)]
for i in linspace(2,n1-1,n1-2):
    m1 = append(m1,(i-0.3175)/(n1+0.365))
m1 = append(m1,0.5**(1./n1))

m2 = [1.-0.5**(1./n2)]
for i in linspace(2,n2-1,n2-2):
    m2 = append(m2,(i-0.3175)/(n2+0.365))
m2 = append(m2,0.5**(1./n2))


f1 = interp1d(m1,ss1)
f2 = interp1d(m2,ss2)


fig, ax = subplots()
ax.tick_params(axis='both',labelsize=14)
ax.set_xlabel(r'sample 1 quantiles',fontsize=16)
ax.set_ylabel(r'sample 2 quantiles',fontsize=16)
ax.plot(f1(m),f2(m),'ko',label='Data')
ax.plot(f1(m),f1(m),'k-')
savefig('q-q_plot.pdf',bbox_inches='tight')
show()
```

8.9 Are two Distributions the Same?

In the previous section we saw how to plot data so as to get an idea about whether data could originate from a certain distribution. $Q-Q$-plots or normality plots are good tools to decide. But they do not offer a statistic which allows us to decide whether two distributuons are different or the same. This deficiency is overcome by the **Kolmogorov-Smirnov test**. Figure 8.7 explains how this test works. We plot the cumulative distribution function of the two distributions which are to be compared vs. their independent variables (the "x" variable). Then the minimum value along the y axis is 0, the maximum is unity. One then computes the maximum difference between the two cumulative distribution functions,

$$D_{n_1,n_2} = \max(|P_{1,n_1} - P_{2,n_2}|).$$

This can be a little tricky if the two samples did not have the same length. In Fig. 8.7 I have tried to solve this by using interpolated values for the two samples.

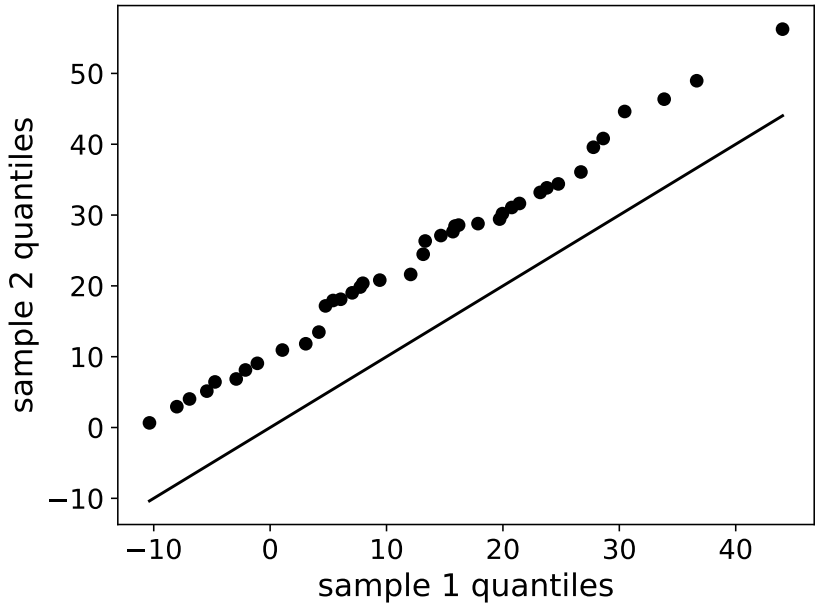


Figure 8.6: An example for a q-q plot. The code which was used to generate it is shown in the listing on page 122.

This introduces some ambiguity, but the results do not differ too much and the effect on the final test described below is negligible. If it isn't, you're up to no good...

We define the null hypothesis, $\mathbb{H}_0$, to be that the two distributions are the same. It is rejected at significance level α if

$$D_{n_1,n_2} > c(\alpha)\sqrt{\frac{n_1+n_2}{n_1 \cdot n_2}}, \quad c(\alpha) = \sqrt{-\frac{1}{2}\ln\left(\frac{\alpha}{2}\right)}.$$

In the examples plotted in Fig. 8.7 the difference is large and the null hypothesis is rejected.

`scipy` offers an implementation of the Kolmogorov-Smirnov test which is summarized below.

TO BE WRITTEN

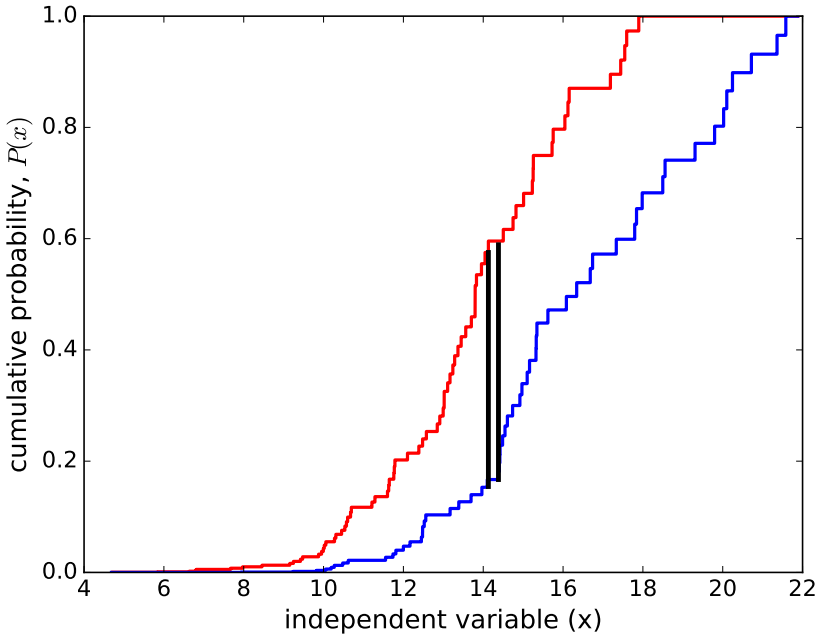


Figure 8.7: Example for a Kolmogorov-Smirnov test. The cumulative probability distribution functions of two samples are plotted vs. their independent variables. The Kolmogorov-Smirnov test statistic is the maximum of the difference of the two cumulative probability distribution functions which is indicated by black lines. This ought to be one single black line, but here I chose to illustrate the difficulty in determining this difference when the two samples don't have the same length. See text for explanantion.

Chapter 9

Fitting a Model to the Data

We are often faced with the problem of determining the values of some parameters of a physical model which underlies our measurements. The need for fitting a model to the data to obtain the underlying parameters arises because we can often not measure the parameters directly. We are in a different situation than when for instance measuring the weights of bags of coffee, or the temperature of a cup of tea. In our field, we often wish to know the exponent of an energy spectrum which is typically assumed to be a power law in energy or energy per mass (velocity squared in the non-relativistic limit), or more complicated, a power law with an exponential roll over,

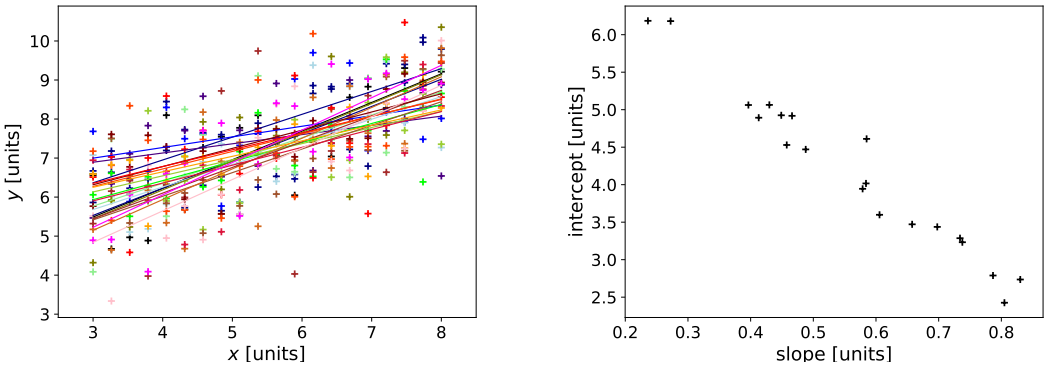
$$J = J_0 \left(\frac{E}{m}\right)^{-\gamma}, \quad \text{or} \quad J = J_0 \left(\frac{E}{m}\right)^{-\gamma} \cdot e^{-E/E_0}.$$

In the first case we need to estimate two parameters, J_0 and γ, while in the second we need to find three. Other examples are the energy calibration of a detector which can be determined by using the Compton edges produced in the detector by X-rays, or using conversion electrons, etc. In this case, we often only need to fit a straight line to the data to obtain a calibration line or the corresponding calibration factors, the gain and offset.

Note that we have already considered a special case of "fitting" when we discussed linear regression in section 7.4, as we will see in the first section 9.1. Before we do so, consider Fig. 9.1. Its left-hand panel shows 20 different "data" sets which were generated from the same underlying linear model function in different colors. The linear model was simply

$$y(x) = 4.5 + x/2,$$

and (y) data were randomized with standard normally distributed errors, $\sigma = 1$. That means that we may imagine at each location x_i a normal probability



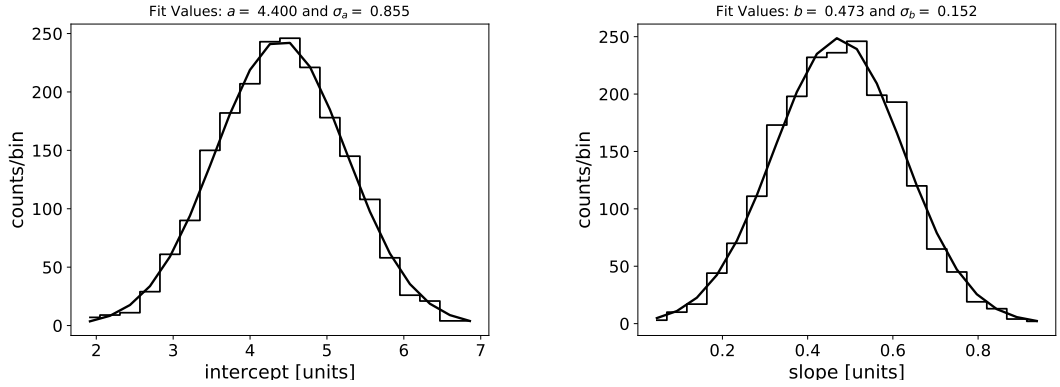
(a) Data and results of fitting a linear model to the data.

(b) Plot of the resulting fitted intercepts vs. slopes.

Figure 9.1: Illustration that fitting can only give a probabilistic answer. Left-hand panel: Using the same underlying model, 20 data sets were generated and fitted. Each set and fit is plotted in the same color. Right-hand panel: Intercepts and slopes resulting from the fits plotted vs. each other. The obvious anticorrelation is discussed in the text.

distribution for data points from which one point is chosen. This is done for every single location yielding 20 different sets of "data" points. Each "data" set was then fitted with a linear function and the resulting fits are shown in the same colors as the "data" set. This plot illustrates clearly that fitting can only give a probabilistic answer. We can't know the "true" underlying parameters, all we can do is to approximate them and give an estimate how well we know them. We can also give an estimate for how far the fitted "model" could lie from the "true" underlying model. This is the subject of this chapter which makes heavy use of *Press et al.* (1989) chapter 14 and of *Richter* (1995). I strongly recommend to also read these sources.

The right-hand panel of Fig. 9.1 shows the intercepts and slopes resulting from the linear fits of the left-hand panel plotted vs. each other. The obvious anti-correlation can be understood the folowing way. Because the intercept and slope are both positive, and the "center of gravity" of the data points lies - by construction - at a higher y value than the intercepts, an increased fitted intercept must go with a smaller slope for the model line to go through the cloud of data points. So despite the fact that we would expect the slope and intercept to be independent of each other they are not. Figure 9.2 shows histograms of the slopes and intercepts of 2000 such fits to artificial "data sets". One can clearly see that they are distributed as one would expect, as a Gaussian distribution centered on their expectation values which we know in this case, because we invented them.



(a) Distribution of fitted intercepts.

(b) Distribution of fitted slopes.

Figure 9.2: Histograms of fitted intercepts and slopes show the expected Gaussian distribution around their central values.

9.1 Linear Regression and/or Fitting a Straight Line to Data

Let us begin with the simple example of fitting a straight line to data points. We assume that we have n data points (x_i, y_i) to which we want to fit a linear model,

$$y(x) = a + b \cdot x,$$

with parameters a and b (for instance, offset and gain of our amplifier chain). To keep matters simple we will - for the time being - assume that the values of the independent variable, x_i, are known exactly, i.e., have no uncertainties. This could, for example, be the known energies of conversion electrons or X-rays. We will, however, allow for errors, σ_i, in the values of the dependent variable, y_i. In the previous example, this would be the detector response to an energy deposit by a conversion electron or an X-ray. Because we have only limited resolution and other possible sources of errors, we allow for uncertainties in $y(x_i)$.

The most common way to fit a line $y(x) = a + b \cdot x$ to the data (x_i, y_i), is to minimize the quantity

$$\chi^2(a,b) = \sum_{i=1}^{n} \left(\frac{y_i - y(x_i)}{\sigma_i} \right)^2 = \sum_{i=1}^{n} \left(\frac{y_i - (a + b \cdot x_i)}{\sigma_i} \right)^2, \tag{9.1}$$

which is called the "chi squared". As we will see later on, this procedure results in a maximum likelihood estimate for the two parameters, a and b if the errors σ_i are normally distributed around the expectation value. If the errors are not normally

distributed, this technique can often still yield good results, but does by no means have to do so!

The minimization of χ^2 is easily understood as minimizing the overall "distance" of the model function (a straight line) to the data. In this case, the "distance" is only in one dimension, the (dependent) y direction. We will see in section 9.8 how to treat errors in both x and y.

We proceed to minimize χ^2 as usual by setting the derivatives of χ^2 with respect to a and b to zero and solving for a and b because these parameters are the ones which minimize the χ^2.

$$0 = \frac{\partial \chi^2}{\partial a} = -2 \sum_{i=1}^{n} \frac{y_i - a - b \cdot x_i}{\sigma_i^2}, \tag{9.2}$$

$$0 = \frac{\partial \chi^2}{\partial b} = -2 \sum_{i=1}^{n} \frac{x_i (y_i - a - b \cdot x_i)}{\sigma_i^2}. \tag{9.3}$$

Obviously, the factors (-2) are irrelevant for the minimization, and we are left with a system of two linear equations for the two parameters, a and b,

$$\begin{aligned} a \cdot \sum_{i=1}^{n} \frac{1}{\sigma_i^2} + b \cdot \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} &= \sum_{i=1}^{n} \frac{y_i}{\sigma_i^2}, \\ a \cdot \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2} + b \cdot \sum_{i=1}^{n} \frac{x_i^2}{\sigma_i^2} &= \sum_{i=1}^{n} \frac{x_i y_i}{\sigma_i^2}. \end{aligned}$$

The sums

$$S \doteq \sum_{i=1}^{n} \frac{1}{\sigma_i^2}, \quad S_x \doteq \sum_{i=1}^{n} \frac{x_i}{\sigma_i^2}, \quad S_y \doteq \sum_{i=1}^{n} \frac{y_i}{\sigma_i^2}, \quad S_{xx} \doteq \sum_{i=1}^{n} \frac{x_i^2}{\sigma_i^2}, \quad S_{xy} \doteq \sum_{i=1}^{n} \frac{x_i y_i}{\sigma_i^2}$$

are easily computed from the data, in fact so easily that you can find linear regression on pocket calculators (should you still have one...). Expressed with these new variables, the system of equations looks even easier to solve,

$$\begin{aligned} a \cdot S + b \cdot S_x &= S_y, \\ a \cdot S_x + b \cdot S_{xx} &= S_{xy}, \end{aligned}$$

and has the solution

$$a = \frac{S_{xx} S_y - S_x S_{xy}}{\Delta}, \tag{9.4}$$

$$b = \frac{S S_{xy} - S_x S_y}{\Delta}, \quad \text{where} \quad \Delta = S S_{xx} - S^2. \tag{9.5}$$

Let us look at these expressions in some more detail. We have assumed that we have an underlying model, $y(x) = a + b \cdot x$, from which we drew a sample of data points,

$$y_i = a + b \cdot x_i + \varepsilon_i, \tag{9.6}$$

where the ε_i are the (unknown) errors in the measurements and are assumed to be normally distributed. Our minimization is thus nothing else but the minimization of

$$\chi^2 = \sum_{i=1}^{n} \delta_i^2 = \sum_{i=1}^{n} \left(y_i - a - b \cdot x_i\right)^2,$$

by solving for the candidate parameters, a and b. There are two things to note here. First, and more importantly, we are mixing up the "true" model parameteres a and b from eq. 9.6 with the best-fit model paramters a and b which we are attempting to find. This is sloppy, but I did not want to introcude new variables which we will never use again at this point. Second, it is wortwhile to note that we are now solving to minimize the sum of the squared **residuals**,

$$\delta_i \doteq y_i - y(x_i) = y_i - a - b \cdot x_i.$$

Let us again take the derivatives of χ^2 with respect to a and b and set them equal to zero in order to minimize,

$$\begin{aligned} 0 = \frac{\partial \chi^2}{\partial a} &= -2 \sum_{i=1}^{n} \left(y_i - a - b \cdot x_i\right), \\ 0 = \frac{\partial \chi^2}{\partial b} &= -2 \sum_{i=1}^{n} x_i \left(y_i - a - b \cdot x_i\right). \end{aligned}$$

We solve for a,

$$\begin{aligned} 0 &= \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} a - \sum_{i=1}^{n} b x_i, \\ n \cdot a &= \sum_{i=1}^{n} y_i - b \sum_{i=1}^{n} x_i, \\ n \cdot a &= n \cdot \bar{y} - bn \cdot \bar{x}, \\ a &= \bar{y} - b \cdot \bar{x}. \end{aligned} \tag{9.7}$$

Solving for b,

$$\begin{aligned}
0 &= \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} a x_i - \sum_{i=1}^{n} b x_i x_i, \\
b\sum_{i=1}^{n} x_i x_i &= \sum_{i=1}^{n} y_i x_i - a\sum_{i=1}^{n} x_i, \\
b\sum_{i=1}^{n} x_i x_i &= \sum_{i=1}^{n} y_i x_i - (\bar{y} - b\cdot\bar{x})\sum_{i=1}^{n} x_i, \\
b\left(\sum_{i=1}^{n} x_i x_i - n\bar{x}^2\right) &= \sum_{i=1}^{n} y_i x_i - n\bar{x}\bar{y}, \\
b\sum_{i=1}^{n}(x_i-\bar{x})^2 &= \sum_{i=1}^{n}(y_i-\bar{y})(x_i-\bar{x}), \\
b &= \frac{\sum_{i=1}^{n}(y_i-\bar{y})(x_i-\bar{x})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}, \qquad (9.8)
\end{aligned}$$

where you need to convince yourself that the second last line is correct[1]!

The reason for going through this seemingly complicated procedure was to show you the following. Note that we can rewrite eq. 9.8 using eqs. 7.3 and 7.7 for the empirical variance and covariance which we discussed in sections 7.2 and 7.4,

$$b = \frac{\mathrm{Cov}_{\mathrm{emp}}[x,y]}{\mathrm{Var}[x]}. \qquad (9.9)$$

In fact, we can also write b using Pearson's correlation coefficient, r_{xy} (eq. 7.8),

$$b = r_{xy}\frac{s_y}{s_x}, \qquad (9.10)$$

where s_x and s_y are the standard deviations of x and y. Note that these expressions are identical to eqs. 9.4 and 9.5 which we found a moment ago. The slope and intercept of the best-fit linear model are determined by the empirical covariance of the data and by the data's average properties. This can be shown even more clearly by rewriting our model function, $y(x) = a + b\cdot x$ using the equations above for a and b,

$$\begin{aligned}
y(x) &= a + b\cdot x, \\
y(x) &= \bar{y} - b\cdot\bar{x} + b\cdot x = \bar{y} + b\cdot(x-\bar{x}), \\
y(x) - \bar{y} &= r_{xy}\frac{s_y}{s_x}\cdot(x-\bar{x}), \\
\frac{y(x)-\bar{y}}{s_y} &= r_{xy}\cdot\frac{x-\bar{x}}{s_x}, \qquad (9.11)
\end{aligned}$$

[1] It is. Show it by working backwards.

which standardizes our function by centering it on $(\bar{x}, \bar{y})$ and normalizes it by dividing by the standard deviations. In this way, there is no intercept, and the slope is given by Pearson's correlation coefficient, eq. 7.8.

Example 9.1. Consider the following code snippets which together generate Figure 9.3.

```
from pylab import *
from scipy import stats
from numpy import random

a = 4.5; b = 0.5; n = 16; err=0.5
random.seed(12345678) #to make this reproducible!
x = linspace(0,10,n)
y = a + b*x + err*random.random(n)

sl, inter, r, p, err = stats.linregress(x, y)
xf = linspace(min(x),max(x),10)
yf = inter+sl*xf

fig, (ax1,ax2) = subplots(2,1,sharex=True,
      gridspec_kw = {'height_ratios':[3, 1]})
ax1.errorbar(x,y,yerr=err*ones(n),fmt='ko',lw=2,label='data')
ax1.plot(xf,yf,'r-',lw=2,label='fit')
ax2.errorbar(x,y-yf,yerr=err*ones(n),fmt='ko',lw=2)
ax2.plot((xmin,xmax),(0,0),'k-',lw=2)
ax2.yaxis.set_ticks((-0.5,0.,0.5))
fig.subplots_adjust(hspace=0)
```

It is a good idea to plot the residuals after one has performed a fit. They can show whether there is a remaining dependence in the data which we did not recognize previously.

9.2 Determining the Errors of the Fit Parameters

Figure 9.3 has a big problem... We have no idea how big the errors are of the slope and the intercept. In fact, the errors in the residuals are nothing else but the errors of the data with no indication of the additional error due to the fitting uncertainty. So how can we determine the errors of the slope and of the intercept? To understand the problem, imagine that we could have drawn another measurement, $y(x_i)$, out of the many possible ones for each x_i (see Fig. 9.1). Had this

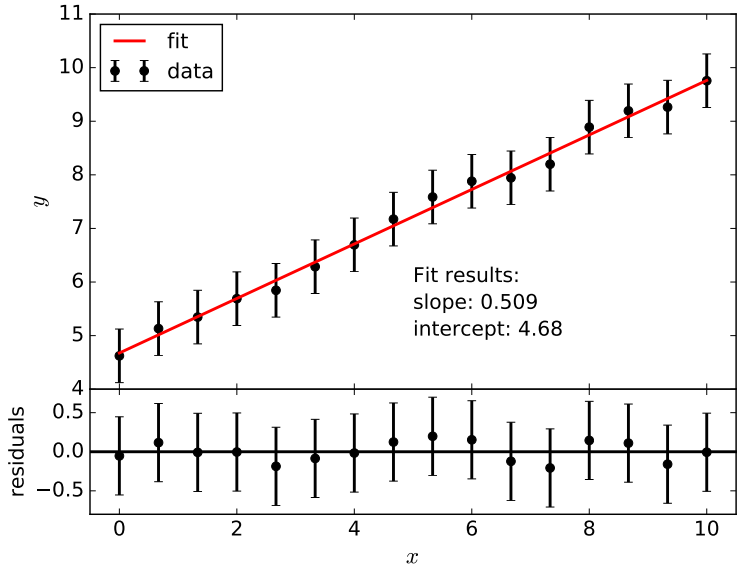


Figure 9.3: An example of a linear fit.

measurement been different, we would have obtained a slightly different estimate for the slope and intercept. Let us begin with estimating the error of the slope by computing its variance,

$$\mathrm{Var}\,[b] = \mathrm{Var}\left[\frac{\sum_{i=1}^{n}(y_i-\bar{y})(x_i-\bar{x})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}\right],$$

where we have inserted b from eq. 9.8 into the "variance operator". The sum in the denominator is constant because all values x_i are fixed, so, using the rules for calculating with the variance discussed in sec. 5.4.1 we can also write

$$\mathrm{Var}\left[\frac{\sum_{i=1}^{n}(y_i-\bar{y})(x_i-\bar{x})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}\right] = \frac{1}{\left(\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^2}\,\mathrm{Var}\left[\sum_{i=1}^{n}(y_i-\bar{y})(x_i-\bar{x})\right].$$

Thus we only need to compute the variance of the following term,

$$\begin{aligned}
\sum_{i=1}^{n}(y_i-\bar{y})(x_i-\bar{x}) &= \sum_{i=1}^{n}(x_i-\bar{x})y_i - \sum_{i=1}^{n}(x_i-\bar{x})\bar{y}, \\
&= \sum_{i=1}^{n}(x_i-\bar{x})y_i - \bar{y}\sum_{i=1}^{n}(x_i-\bar{x}), \\
&= \sum_{i=1}^{n}(x_i-\bar{x})y_i - \bar{y}(\sum_{i=1}^{n}x_i - \sum_{i=1}^{n}\bar{x}), \\
&= \sum_{i=1}^{n}(x_i-\bar{x})y_i, \\
&= \sum_{i=1}^{n}(x_i-\bar{x})(a+b\cdot x_i+\varepsilon_i), \\
&= a\sum_{i=1}^{n}(x_i-\bar{x}) + b\sum_{i=1}^{n}(x_i-\bar{x})x_i + \sum_{i=1}^{n}(x_i-\bar{x})\varepsilon_i.
\end{aligned}$$

We can split the variance of this expression into the sum of the variances of the three terms because they are independent[2]. But taking the variance of these three terms, only the last will survive because the two first terms are constants, i.e., have a vanishing variance. Thus, the variance is given by,

$$\mathrm{Var}\left[b\right] = \frac{1}{\left(\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^2}\,\mathrm{Var}\left[\sum_{i=1}^{n}(x_i-\bar{x})\varepsilon_i\right].$$

We are not quite done yet. Again, $\sum_{i=1}^{n}(x_i-\bar{x})$ inside the variance is a constant, and can be taken out of the variance because of $\mathrm{Var}\left[aX\right] = a^2\,\mathrm{Var}\left[X\right]$. Finally, $\mathrm{Var}\left[\sum_{i=1}^{n}\varepsilon_i\right] = \sigma^2$ where σ^2 is the "spread" of the individual measurements around the underlying physical model. Thus, using the definition of the empirical standard deviation (eq. 7.4), we now have the expression for the error of the slope,

$$\sigma_b = \sqrt{\frac{\sigma^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}} = \frac{\sigma}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}} = \frac{\sigma}{\sqrt{n-1}\sigma_x}. \tag{9.12}$$

[2]See the properties of the variance on page 43.

The derivation of this result is more easily understood if we define the quantities $X_i \doteq x_i - \bar{x}$ and $Y_i \doteq y_i - \bar{y}$. With these quantities we now have,

$$\begin{aligned}
\mathrm{Var}\left[\sum_{i=1}^{n} X_i Y_i\right] &= \mathrm{Var}\left[X_1 Y_1 + X_2 Y_2 + \ldots X_n Y_n\right], \\
&= X_1^2 \,\mathrm{Var}\left[Y_1\right] + X_2^2 \,\mathrm{Var}\left[Y_2\right] + \ldots X_n^2 \,\mathrm{Var}\left[Y_n\right], \\
&= \sum_{i=1}^{n} X_i^2 \,\mathrm{Var}\left[Y_i\right], \\
&= \sigma_y^2 \sum_{i=1}^{n} X_i^2.
\end{aligned}$$

In preparation of the following determination of the error of the intercept, we determine the error of the model function, $y(x) = a + bx$, where $a = \bar{y} - b \cdot \bar{x}$,

$$\mathrm{Var}\left[y(x)\right] = \mathrm{Var}\left[\bar{y} + b(x - \bar{x})\right] = \mathrm{Var}\left[\bar{y}\right] + (x - \bar{x})^2 \,\mathrm{Var}\left[b\right].$$

With

$$\mathrm{Var}\left[\bar{y}\right] = \mathrm{Var}\left[\frac{\sum_{i=1}^{n} y_i}{n}\right] = \frac{1}{n^2}\,\mathrm{Var}\left[\sum_{i=1}^{n} y_i\right] = \frac{n\sigma^2}{n^2} = \frac{\sigma^2}{n},$$

and the square of eq. 9.12 to convert to variance, we have

$$\mathrm{Var}\left[y(x)\right] = \mathrm{Var}\left[\bar{y} + b(x - \bar{x})\right] = \frac{\sigma^2}{n} + \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}(x - \bar{x})^2, \tag{9.13}$$

$$\sigma_{y(x)} = \sigma\sqrt{\frac{1}{n} + \frac{(x - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}. \tag{9.14}$$

The error of the intercept now follows trivially by considering eqs. 9.13 and 9.14 at $x = 0$,

$$\mathrm{Var}\left[a\right] = \frac{\sigma^2}{n} + \frac{\sigma^2(\bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \quad \text{and} \quad \sigma_a = \sigma\sqrt{\frac{1}{n} + \frac{(\bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}. \tag{9.15}$$

Finally, we also compute the error for a prediction of a new measurement y_{pred} at a (possibly new) value x. Because measurements are spread around $y(x)$ normally according to $\mathcal{N}(y(x), \sigma)$, we have to add this variance to our result for

the variance of $y(x)$, given in eq. 9.13. Thus,

$$\begin{aligned} \mathrm{Var}\left[y_{\mathrm{pred}}\right] &= \mathrm{Var}\left[y(x)\right] + \sigma^2 = \frac{\sigma^2}{n} + \frac{\sigma^2 (x_i - \bar{x})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} + \sigma^2, \\ &= \sigma^2 \left(1 + \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2}\right), \qquad (9.16) \\ \sigma_{y_{\mathrm{pred}}} &= \sigma \sqrt{1 + \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2}}. \qquad (9.17) \end{aligned}$$

Example 9.2. To determine the work function, W, of a metal surface, we can irradiate it with photons of known energies, $E_\gamma = h \cdot \nu$ and measure the kinetic energy of the electrons which are emitted by the material, in other words, by exploiting the photo effect. The energy of the electrons is then $E_e = E_\gamma - W$ and can easily be measured by applying a retarding potential or using an energy analyzer (E/q-filter). If we use photons with different energies, e.g., by applying narrow-band filters, we will measure different electron energies, E_e, at different E_γ. Plotting E_e vs. E_γ will give a straight line. Extrapolating it to $E_e = 0$ will give us the work function of the metal. The error in this quantity can now be computed given the relations we derived in this section. **TODO**

Summary 9.1. The following table provides a summary of the standard errors of the various properties of the fitted model function. The two predictions are especially important, as we will see in the next section. It should not come as a surprise to you that the prediction of a new measurement has a substantially larger error than the prediction of the mean (the model function).

std. error of the slope	$\sigma_b = \sqrt{\frac{\sigma^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}} = \frac{\sigma}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}} = \frac{\sigma}{\sqrt{n-1}\sigma_x}$,
std. error of intercept	$\sigma_a = \sigma\sqrt{\frac{1}{n} + \frac{(\bar{x})^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}}$,
std. error of predicted mean	$\sigma_{y(x)} = \sigma\sqrt{\frac{1}{n} + \frac{(x-\bar{x})^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}}$,
std. error of predicted value (new measurement)	$\sigma_{y_{\mathrm{pred}}} = \sigma\sqrt{1 + \frac{1}{n} + \frac{(x-\bar{x})^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}}$.

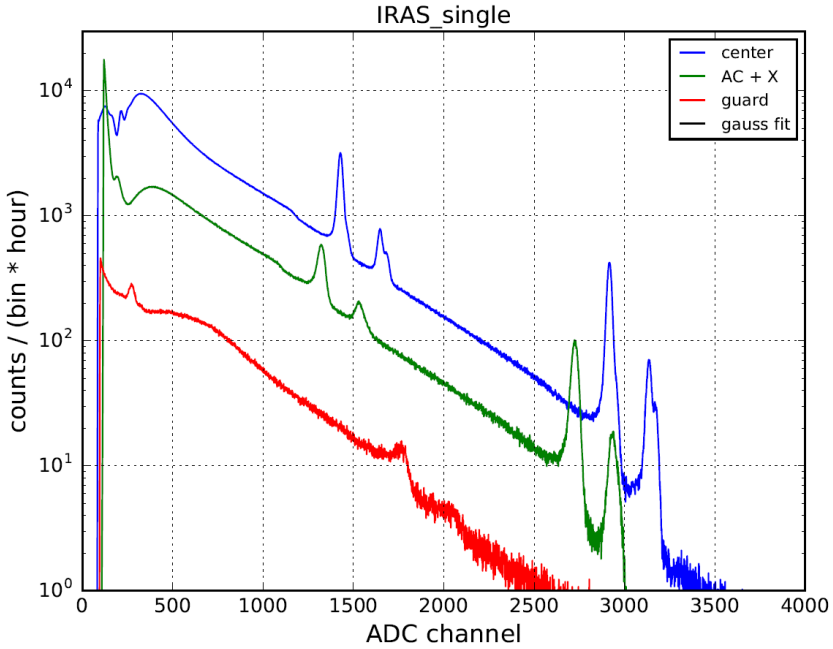


Figure 9.4: ^{207}Bi spectrum acquired with an IRAS detector. From *Knappmann* (2016).

9.3 Presenting your Results: A Calibration Line

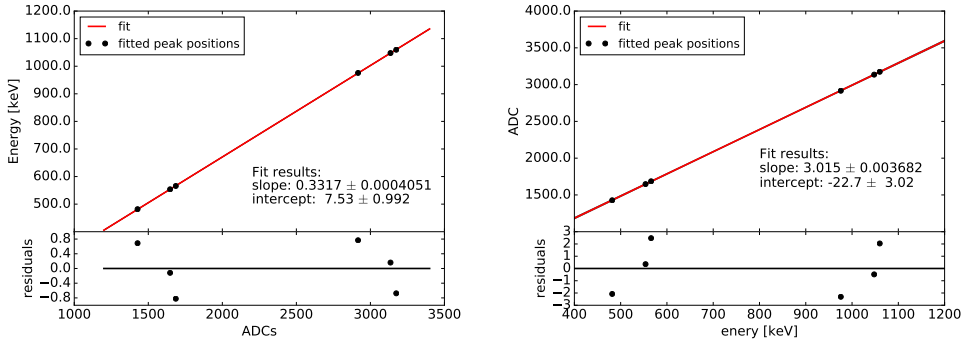
^{207}Bi decays by electron capture and has four (six) prominent lines at 481.7 keV (1.54%), 553.8 keV (0.44%), (565.8 keV (0.442%)), 975.7 keV (7.1%), and 1047.8 keV (1.8%), (1059.8 keV (0.44%)), where intensities are given in parenthesis. This source is often used to calibrate solid state detectors because these energetic electrons are hardly affected by the detector's dead layer and air, but (some) stop in 300 or 500 μm Si PIPS detectors which are used in many of our instruments. Lines shown in parenthesis are not necessarily fully resolved in instruments. Figure 9.4 which is taken from *Knappmann* (2016) shows a spectrum acquired with an IRAS detector used for the Lunar Lander Neutron and Dosimetry (LND) experiment on China's Chang'E4 mission to the far side of the Moon. One can easily see the four prominent lines and recognizes the double peaks which are due to the two only barely resolved less intense lines in the data of the central segment of the IRAS detector.

The peaks and double peaks can be fitted with a model function which involves Gaussian peak shapes and exponential backgrounds locally around the individual peaks. We will see how to perform similar fits later on in this chapter. From such a fit, *Knappmann* (2016) arrived at the ADC values given in Tab. 9.1.

The data can be plotted and fitted with a straight line as shown in the previous

energy [keV]	peak position [ADC]
481.7	1427.5
553.8	1647.3
565.8	1685.6
975.7	2916.6
1047.8	3135.8
1059.8	3174.5

Table 9.1: Values of fitted peak positions (in ADCs) and the corresponding energies (in keV) of the ^{207}Bi conversion electrons. After *Knappmann* (2016).



(a) LND ^{207}Bi calibration data plotted as energy vs. ADC values.

(b) LND ^{207}Bi calibration data plotted as ADC values vs. energy.

Figure 9.5: Presentation of calibration results. The 99.9% confidence bands for the mean and new measurements are invisible (because the calibration is so good).

section or using `scipy` 's `curve_fit` routine. The results are presented in Fig. 9.5.

Actually, I meant to show you how to now plot a real calibration line such as that shown in Fig. 2.4 in Ch. 2. But it turned out that the calibration of LND's detectors is so good that even the 99.9% confidence band for the mean and the corresponding prediction band for new measurements are invisible in Fig. 9.5. So here's another example with some fake calibration data, as shown in Fig. 2.4. That's why I've replotted that figure with residuals in Fig. 9.6.

How do we determine the confidence and prediction bands shown in that figure? Basically, we want to give a confidence interval for the sample mean, $\bar{x}$, at a given value of the independent (control) variable to agree with the population mean, μ, at that location. As we showed in sec. 8.4, this is exactly the situation which is treated by Student's t. We repeat eq. 8.21 here for convenience because it shows

how to report the mean together with its $1-\alpha$ confidence interval.

$$\mu = \bar{x} \pm t_{\alpha/2,n-1}\frac{s}{\sqrt{n}}, \tag{8.21}$$

where s is the empirical standard deviation and n the number of samples taken for this mean. The multiplication of the empirical standard deviation by Student's t takes care of the additional uncertainty that we have because we had to estimate the population mean from a sample mean. A summary is given in summary 8.4.

In summary 8.4 we can also remind ourselves that the prediction interval for a new measurement is obtained by quadratically adding the standard error of a measurement to the last term in eq. 8.21,

$$\mu = \bar{x} \pm t_{\alpha/2,n-1}s\sqrt{1+\frac{1}{n}}. \tag{8.22}$$

so this is then what we need to do here too and that is what we have plotted in Fig. 9.6 using the errors of the mean (eq. 9.14) and of a new measurement (9.16) both multiplied by Student's t for the relevant confidence level (here 95%) and degrees of freedom.

Again, the reason for this additional "safety factor", $t_{\alpha/2,n-1}$, is explained in sec. 8.4 and in summary 8.4. Basically, it is here to account for our determining all fit parameters from the data without having any knowledge of the underlying "true" model parameters.

Definition 9.1. A $(1-\alpha)$ **confidence band** is a band around the model **for the mean** which has a probability α of being wrong (in the sense of hypothesis testing).

Definition 9.2. A $(1-\alpha)$ **prediction band** is a band around the model **prediction for a new measurement** which has a probability α of observing a new measurement outside of this band.

Careful inspection of Fig. 9.6 reveals that the shaded areas are slightly curved and are narrowest around the middle of the plot and get wider the farther away we move from the middle. This is not just an impression, but is the result of the term $(x-\bar{x})^2$ in the numerator of the last terms in the sqare root of eqs. 9.14 and 9.17. Thus, if you want your calibration to be especially good in a certain region then you should ensure that you fit around that region. Your fit will always be the least certain at the boundaries of your fit region, so take that into account! You can also see this more clearly by inspecting once more, Fig. 9.1 in the introduction to this chapter.

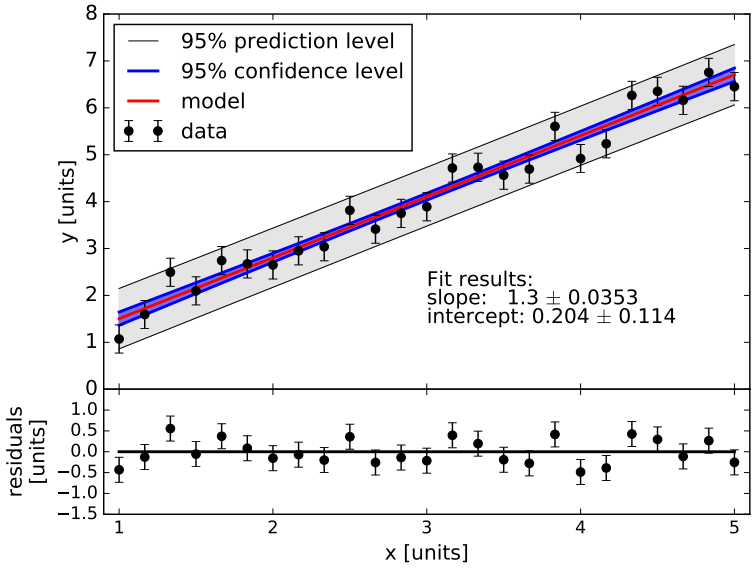


Figure 9.6: An invented calibration line showing the 95% confidence band for the mean and new measurements as well as the residuals.

Summary 9.2. To plot a $(1-\alpha)$ **confidence band for the mean** and a $(1-\alpha)$ **prediction band for new measurements**, use the following expressions which are valid for a linear fit $y(x) = a + b \cdot x$ only:

conf. band for mean $$c(x) = y(x) \pm t_{1-\alpha/2,n-2} \cdot \sigma \sqrt{\frac{1}{n} + \frac{(x-\bar{x})^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}},$$

pred. band for new measurement $$p(x) = y(x) \pm t_{1-\alpha/2,n-2} \cdot \sigma \sqrt{1 + \frac{1}{n} + \frac{(x-\bar{x})^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}},$$

where n is the number of data points and $t_{1-\alpha/2,n-2}$ is Student's t, and σ is the error of the measurements on the y axis. Student's t needs to be calculated for $n-2$ degrees of freedom because the linear fit "uses up" two degrees of freedom.

9.4 Linear Least Squares Fitting of a Model

Quite often, a simple linear model can not explain the data and we need to fit a more complicated model to the data. This could be a polynomial fit to a complicated curve or an exponential fit to the activity of a radioactive source. The key point in this section is that the model depends on the model parameters in a linear manner. A (crazy) example could be

$$y(x) = a + b \cdot x + c \cdot \sin((\pi/2)x) + d \cdot \exp(-x). \tag{9.18}$$

As you see, the model, $y(x)$, depends linearly on the parameters a, b, c, and d although the so-called **basis functions** (1, x, $\sin((\pi/2)x)$, and $\exp(-x)$) are nonlinear. In this section, I will present fitting techniques for model functions of the form

$$y(x, \mathbf{a}) = \sum_{j=1}^{m} a_j F_j(x), \tag{9.19}$$

where $F_j(x)$ are arbitrary functions of x and $y(x, \mathbf{a})$ depends linearly on the model paramters, a_j. The more general case where the model function can depend nonlinearly on the model parameters is treated in Sec. 9.6.

We again determine the optimal parameters by minimizing the χ^2,

$$\chi^2(\mathbf{a}) \doteq \sum_{i=1}^{n} \frac{(y_i - y(x_i, \mathbf{a}))^2}{\sigma_i^2}, \tag{9.20}$$

where σ_i are the errors of the measurements where known. If we don't know them or all σ_i are equal, they can be set equal to one, $\sigma_i = 1$, $\forall i$. For the following treatment, note that there are n data points and m model parameters. We can express χ^2 using elementary linear algebra by noting that the data points y_i can be viewed as an n-dimensional vector, the parameters as an m-dimensional one, the model function, $y(x_i, \mathbf{a})$, evaluated at x_i is in turn also a n-dimensional vector. The m basis functions, F_j, are also evaluated at all x_i and so $F_j(x_i)$ can be expressed as a $n \times m$ matrix, F_{ij}. With this notation, eq. 9.19 is transformed into

$$y(x_i, \mathbf{a}) = \sum_{j=1}^{m} F_{ij} a_j, \quad i \in \{1, 2, \ldots, n\}, \quad \text{or in matrix form} \quad \mathbf{y} = \mathbf{F}\mathbf{a}. \tag{9.21}$$

We also define the normalized vector $\mathbf{b}$ and normalized matrix $\mathbf{A}$ by their components,

$$b_i \doteq \frac{y_i}{\sigma_i}, \quad \text{and} \quad A_{ij} \doteq \frac{F_{ij}}{\sigma_i}, \tag{9.22}$$

where A_{ij} is often called the **design matrix** for the least-squares fit (*Press et al.*, 1989). Using these definitions, we can write χ^2 (eq. 9.20) as a matrix multiplication,

$$\chi^2 = (\mathbf{b} - \mathbf{A}\mathbf{a})^T(\mathbf{b} - \mathbf{A}\mathbf{a}), \tag{9.23}$$

where the ^{T} signifies the transpose of the matrix and so ensures that we get a meaningful result. Taking the derivative with respect to the parameters and setting equal to zero to minimize χ^2,

$$\begin{aligned}
\frac{\partial \chi^2}{\partial \mathbf{a}} &= \frac{\partial}{\partial \mathbf{a}}(\mathbf{b} - \mathbf{A}\mathbf{a})^T(\mathbf{b} - \mathbf{A}\mathbf{a}) = 0, \\
&= \left(\frac{\partial}{\partial \mathbf{a}}(\mathbf{b} - \mathbf{A}\mathbf{a})^T\right)(\mathbf{b} - \mathbf{A}\mathbf{a}) + (\mathbf{b} - \mathbf{A}\mathbf{a})^T\left(\frac{\partial}{\partial \mathbf{a}}(\mathbf{b} - \mathbf{A}\mathbf{a})\right), \\
&= -\underbrace{\mathbf{A}^T}_{\text{matrix}}\underbrace{(\mathbf{b} - \mathbf{A}\mathbf{a})}_{\text{vector}} - \underbrace{(\mathbf{b} - \mathbf{A}\mathbf{a})^T}_{\text{vector}}\underbrace{\mathbf{A}}_{\text{matrix}} \\
0 &= 2\mathbf{A}^T(\mathbf{b} - \mathbf{A}\mathbf{a}), \\
(\mathbf{A}^T\mathbf{A})\mathbf{a} &= \mathbf{A}^T\mathbf{b},
\end{aligned} \tag{9.24}$$

where we have divided by 2 in line 5. We can solve eq. 9.24 for the model parameters, $\mathbf{a}$,

$$\mathbf{a} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} = \mathbf{C}\mathbf{A}^T\mathbf{b}, \quad \text{or} \quad a_j = \sum_{j=1}^{m} C_{jk} \sum_{i=1}^{n} y_i \, \frac{F_k(x_i)}{\sigma_i^2}, \tag{9.25}$$

where $\mathbf{C} = (\mathbf{A}^T\mathbf{A})^{-1}$ is the **covariance matrix** of the problem. It describes the covariance of the parameters of the model, as we will see momentarily. This means that we need to expect that the errors or uncertainties in the different parameters may be correlated, which should not come as a surprise to you.

The variances of the parameters can be estimated using the usual law of error propagation,

$$\sigma(a_j)^2 = \sum_{i=1}^{n} \left(\frac{\partial a_j}{\partial y_i}\right)^2 \sigma_i^2,$$

where, as usual, σ_i are the errors of the dependent variable or measurements, y_i. We compute the partial derivatives of the parameters with respect to y_i,

$$\frac{\partial a_j}{\partial y_i} = \sum_{k=1}^{m} C_{jk} \frac{F_k(x_i)}{\sigma_i^2},$$

because C_{jk} and $\mathbf{A}$ are independent of y_i. Inserting this expression into the error propagation, we obtain

$$\sigma^2(a_j) = \sum_{k=1}^{m}\sum_{l=1}^{m} C_{jk}C_{jl}\left[\sum_{i=1}^{n} \frac{F_k(x_i)F_l(x_i)}{\sigma_i^2}\right]. \tag{9.26}$$

The final term in eq. 9.26 is nothing else than $\mathbf{A}^T\mathbf{A}$ and thus the inverse of C_{kl}, thus reducing eq. 9.26 to

$$\sigma^2(a_j) = C_{jj}. \tag{9.27}$$

This means that the variances of the model parameters are given by the diagonal elements of the covariance matrix, C_{jj}. Unsurprisingly, the covariances among the model parameters are given by the off-diagonal elements of the covariance matrix. An example for a general linear least squares fit using the model function given in the introduction to this section in eq. 9.20 is shown in Fig. 9.7. The following parameters were used to generate the data: $a = 1.0$, $b = 0.5$, $c = 1.0$, and $d = 10.0$, $\sigma_i = 0.5$. I used `scipy` 's `curve_fit` to fit the model to the data.

```
from scipy.optimize import curve_fit
a = 1.; b = .5; c=1.; d=10.; dy = 0.5

def func(x,a,b,c,d):
    return a + b*x + c*sin(pi/2*x) + d*exp(-x)

n = 51
x = linspace(0.,10,n)
y = func(x,a,b,c,d) + random.normal(0.,dy,n)
initial_values = array([1.5,0.3,2.,5.])
popt, pcov = curve_fit(func,x,y,p0=initial_values)
print('a = {} +/- {}'.format(popt[0], sqrt(pcov[0,0])))
```

9.5 Confidence and Prediction Bands for General Least Squares Fits

But wait! Where are the confidence and prediction bands in Fig. 9.7? How should they be computed for the more general case considered in the previous section 9.4? Luckily, it turns out that we have already done most of the work for this! Remember the model function,

$$y(x, \mathbf{a}) = \sum_{j=1}^{m} F_{ij} a_j, \quad i \in \{1, 2, \ldots, n\}, \quad \text{or in matrix form} \quad \mathbf{y} = \mathbf{F}\mathbf{a},$$

as given in eq. 9.21. Using this, we can compute the errors of $y(x, \mathbf{a})$ by

$$\delta y(x, \mathbf{a}) = \sum_{j=1}^{m} \delta a_j F_j(x).$$

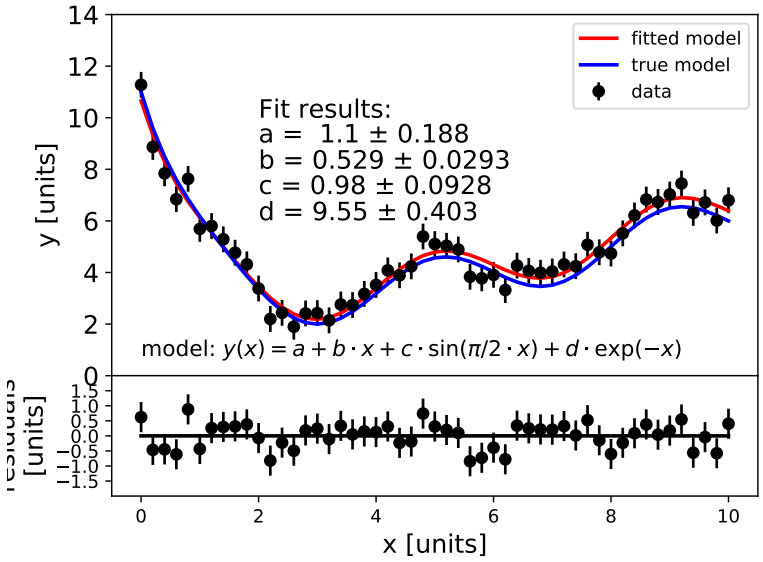


Figure 9.7: Example of a general linear least squares fit with the model function given in eq. 9.20.

Now the covariance $\sigma_y^2(x, x') \doteq \langle \delta y(x) \delta y(x') \rangle$ is given by

$$\sigma_y^2(x, x') = \sum_{j=1}^{m} \sum_{k=1}^{m} \langle \delta a_j \delta a_k \rangle \, F_j(x) F_k(x') = \sum_{j=1}^{m} \sum_{k=1}^{m} C_{jk} F_j(x) F_k(x'). \tag{9.28}$$

Note that this expression shows that the errors of the model function are correlated between x and x', unlike the measurement errors which are independent of each other! Note also, that this result is independent of the model parameters, a_j, and is valid for any values x and x', not only the data points, x_i. In order to be able to plot the confidence and prediction bands, we only use $\sigma_y^2(x, x) = \sigma_y^2(x)$,

$$\sigma_y^2(x) = \sum_{j=1}^{m} \sum_{k=1}^{m} C_{jk} F_j(x) F_k(x) = \mathbf{x}^T(x) \mathbf{C} \mathbf{x}(x), \tag{9.29}$$

where $\mathbf{x}^T(x) = (F_1(x), F_2(x), \ldots, F_m(x))$. Before we proceed to plot this in another example, let us briefly investigate the consequences of eq. 9.29.

Consider the weighted mean of $\sigma_y^2(x)$ averaged over all points $x = x_1, \ldots, x_n$,

$$\frac{1}{n} \sum_{i=1}^{n} \frac{\sigma_y^2(x_i)}{\sigma_i^2} = \frac{1}{n} \sum_{j=1}^{m} \sum_{k=1}^{m} C_{jk} \sum_{i=1}^{n} \frac{F_j(x_i)}{\sigma_i} \frac{F_k(x_i)}{\sigma_i}.$$

Remembering the definition of $\mathbf{A} \doteq F_j(x_i)/\sigma_i$ in eq. 9.22, we see that the sum over i is nothing else than a matrix multiplication,

$$\sum_{i=1}^{n} \mathbf{A}_{ij}\mathbf{A}_{ik} = (\mathbf{A}^T\mathbf{A})_{kj} = (\mathbf{C})^{-1}_{kj},$$

i.e., the inverse of the covariance matrix. Thus we arrive at the following result,

$$\frac{1}{n}\sum_{i=1}^{n}\frac{\sigma_y^2(x)}{\sigma_i^2} = \frac{1}{n}\sum_{j=1}^{m}\sum_{k=1}^{m} C_{jk}(\mathbf{C})^{-1}_{kj} = \frac{1}{n}\sum_{j=1}^{m} I_{jj} = \frac{m}{n}. \tag{9.30}$$

In the case of constant errors, independent of x, i.e., $\sigma_i = \sigma = \text{const.}$, we obtain

$$\bar{\sigma}_y^2 = \frac{1}{n}\sum_{i=1}^{n}\sigma_y^2(x_i) = \frac{m}{n}\sigma^2.$$

In other words, the average variance of the mean of the model predition is $(m/n)\sigma_y^2$, a factor of m larger than the "ordinary" variance of the mean we normally use when averaging data. We see here clearly that the number of degrees of freedom which we use in our model comes at a cost, namely the cost of loss of predicitve power. In yet other words, the more model parameters you need to obtain a good fit, the larger will be the resulting errors of your model function! This certainly drives home the point that you need to find a good model, best a good physical model, with as few parameters as needed.

OK, so now let's use these results to plot a confidence and prediction band in Fig. 9.7. The following code snippets define the basis functions and implementation of the confidence band. The implementation is shown in Fig. 9.8.

```
def func(x,a,b,c,d):
    return a*f0(x) + b*f1(x) + c*f2(x) + d*f3(x)

def f0(x):  #basis function
    return ones(len(x))

def f1(x):  #basis function
    return x

def f2(x):  #basis function
    return sin(pi/2*x)

def f3(x):  #basis function
    return exp(-x)
```

Note that we have explicitly split out the basis functions, F_j, and defined the model function as the sum of the products of the parameters, a_j, and basis functions, F_j. This allows us to access the basis functions individually when we want to calculate the confidence band (eq. 9.29). This is done in the following code snippet.

```
def conf_band(x,yerr,popt,pcov,signif):
    n = len(x)
    np = len(popt)
    f = []
    f = append(f,f0(x,popt))
    f = append(f,f1(x,popt))
    f = append(f,f2(x,popt))
    f = append(f,f3(x,popt))
    f = reshape(f,(np,n))
    d = zeros(n)
    for j in linspace(0,np-1,np):
        for k in linspace(0,np-1,np):
            d += pcov[j,k]*f[j]*f[k]  #basis functions!
    alpha = 1.-signif/2
    tval = t.ppf(alpha,n-np)
    cb = tval*sqrt(d)              #conf band for model func
    pb = tval*sqrt(yerr**2 + d) #pred band for new meas
    return cb, pb
```

To plot the confidence and prediciton bands, we use `matplotlib`'s `filled_between` command,

```
signif = 0.05
cb, pb = conf_band(x,ye,popt,pcov,signif)
ax1.fill_between(x,model_data+pb, model_data-pb,
                 color='grey',alpha=0.2)
ax1.fill_between(x,model_data+cb, model_data-cb,
                 color='blue',alpha=0.5)
```

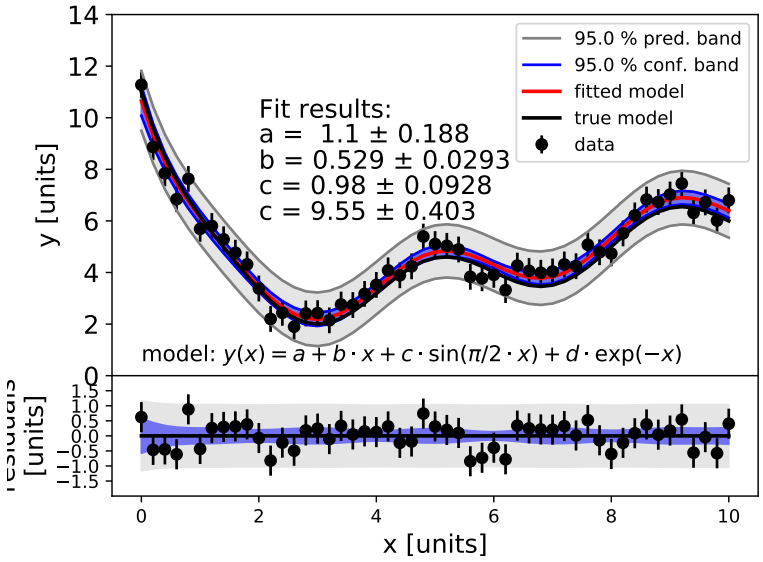


Figure 9.8: The same as Fig. 9.7, but now with confidence and prediction bands.

Summary 9.3. In the general linear least-squares fitting procedure, the error of the model is given by eq. 9.29,

$$\sigma_y^2(x) = \sum_{j=1}^{m}\sum_{k=1}^{m} C_{jk}F_j(x)F_j(x) = \mathbf{x}^T(x)\mathbf{C}\mathbf{x}(x),$$

where C_{jk} is the covariance matrix which is returned by most fitting routines and the $F_j(x)$ are the basis functions which define the model function,

$$y(x, \mathbf{a}) = \sum_{j=1}^{m} F_j(x)a_j.$$

The weighted average variance of the model function, $\bar{\sigma}_y^2 = (m/n) \cdot \sigma^2$ goes as m/n where m is the number of model parameters (degrees of freedom) and n is the number of data points. This tells you to be careful when trying to "improve" your model by adding new parameters. Its predicitve power declines with additional parameters!

9.6 Non-linear Fitting of a Model to Data

Sofar we have treated linear models or linear fits. While the model function used in Sec. 9.4 (eq. 9.18) looked awfully non-linear, it is in fact linear in all the model parameters. This is what is meant with a linear model. We are now going to treat models in which this is no longer the case. One of the most frequent such models is the Gaussian,

$$y(x) = \frac{a}{\sqrt{2\pi c^2}} \exp\left(-\frac{(x-b)^2}{2c^2}\right).$$

Clearly, the parameters b and c appear non-linearly in this example. Note however, that some apparently non-linear problems can be transformed into a linear problem, an example is

$$\text{“non-linear”} \quad y(x) = a\exp(-bx) \quad \xrightarrow{\log} \quad \log(y(x)) = c - bx \quad \text{is linear.}$$

In truly non-linear problems, the model function can be written as

$$y(x) = y(x, \mathbf{a}) = \sum_{j=1}^{m} a_j F_j(x, \mathbf{a}_{jk}), \tag{9.31}$$

in other words, the parameters can appear non-linearly in the arguments of the (non-linear) basis functions. Fitting will still proceed to minimize the χ^2,

$$\chi^2 = \sum_{i=1}^{n} \frac{(y_i - y(x, \mathbf{a}))^2}{\sigma_i^2} = \sum_{i=1}^{n} \frac{(y_i - \sum_{j=1}^{m} a_j F_j(x, \mathbf{a}_{jk}))^2}{\sigma_i^2}.$$

This looks rather complicated and is only shown here to motivate the following considerations. In the linear least squares procedures we could calculate the optimal parameters by linear algebra, by solving a system of linear equations. This is no longer possible when we are faced with non-linear problems. But not all hope is lost! Consider the following: We are looking for the set of parameters, $\hat{\mathbf{a}}$ which minimizes the χ^2 of the problem. If we succeed in finding a reasonably good initial guess for the model parameters, $\mathbf{a}$, wich is not too far from $\hat{\mathbf{a}}$, then we can linearize the model function in this vicinity using a Taylor expansion around $\hat{\mathbf{a}}$,

$$\begin{aligned} y(x, \mathbf{a}) &= y(x, \hat{\mathbf{a}}) + \sum_{j=1}^{m} \left.\frac{\partial y(x, \mathbf{a})}{\partial a_j}\right|_{\hat{\mathbf{a}}} (a_j - \hat{a}_j), \\ &= y(x, \hat{\mathbf{a}}) + (\mathbf{a} - \hat{\mathbf{a}})^T \mathbf{d}(x, \hat{\mathbf{a}}), \qquad (9.32) \\ \text{where} \quad d_j(x, \hat{\mathbf{a}}) &\doteq \left.\frac{\partial y(x, \mathbf{a})}{\partial a_j}\right|_{\hat{\mathbf{a}}}. \end{aligned}$$

Similarly, we can write the χ^2

$$\begin{aligned}\chi^2(\mathbf{a}) &= \chi^2(\hat{\mathbf{a}}) + \sum_{j=1}^{m} \left.\frac{\partial \chi^2(\mathbf{a})}{\partial a_j}\right|_{\hat{\mathbf{a}}} (a_j - \hat{a}_j) + \sum_{j=1}^{m}\sum_{k=1}^{m} \left.\frac{\partial^2 \chi^2(x,\mathbf{a})}{\partial a_j \partial a_k}\right|_{\hat{\mathbf{a}}} (a_j - \hat{a}_j)(a_k - \hat{a}_k) + \ldots, \\ &\approx \chi^2(\hat{\mathbf{a}}) + \sum_{j=1}^{m}\sum_{k=1}^{m} \left.\frac{\partial^2 \chi^2(x,\mathbf{a})}{\partial a_j \partial a_k}\right|_{\hat{\mathbf{a}}} (a_j - \hat{a}_j)(a_k - \hat{a}_k), \qquad (9.33) \\ &= \chi^2(\hat{\mathbf{a}}) + (\mathbf{a} - \hat{\mathbf{a}})^T \mathbf{H}(\hat{\mathbf{a}})(\mathbf{a} - \hat{\mathbf{a}}), \qquad (9.34)\end{aligned}$$

where $\chi^2(\hat{\mathbf{a}})$ is in fact $\chi^2_{\min}$ because that is what $\hat{\mathbf{a}}$ is defined as. The neglection of the linear term in the second line happens for exactly that reason too, it vanishes because it is evaluated at $\hat{\mathbf{a}}$ which minimizes χ^2. The Hessian is defined as

$$\mathbf{H}(\hat{\mathbf{a}}) = h_{jk}(\hat{\mathbf{a}}) = \frac{1}{2} \left.\frac{\partial^2 \chi^2(\mathbf{a})}{\partial a_j \partial a_k}\right|_{\hat{\mathbf{a}}} = \sum_{i=1}^{n} \left.\frac{\partial y(x_i,\mathbf{a})}{\partial a_j}\right|_{\hat{\mathbf{a}}} \left.\frac{\partial y(x_i,\mathbf{a})}{\partial a_k}\right|_{\hat{\mathbf{a}}},$$

where the second step is not obvious, but follows from performing the partial derivatives on

$$\chi^2(\mathbf{a}) = \sum_{i=1}^{n} \frac{\left[y_i - \left(y(x,\hat{\mathbf{a}}) + \sum_{j=1}^{m} \left.\frac{\partial y(x,\mathbf{a})}{\partial a_j}\right|_{\hat{\mathbf{a}}} (a_j - \hat{a}_j)\right)\right]^2}{\sigma_i^2},$$

and noting that all quantities involving only $\hat{\mathbf{a}}$ (and not $\mathbf{a}$) are constant, as well as y_i. The above expression for the Hessian then follows.

Following *Richter* (1995) we now define

$$A_{ij}(\hat{\mathbf{a}}) \doteq \frac{1}{\sigma_i} \left.\frac{\partial y(x_i,\mathbf{a})}{\partial a_j}\right|_{\hat{\mathbf{a}}} = \frac{d_j(x_i,\hat{\mathbf{a}})}{\sigma_i},$$

from which we have

$$\mathbf{H}(\hat{\mathbf{a}}) = \mathbf{A}^T\mathbf{A},$$

which translates exactly to the expressions in the last section 9.4 for the linear least squares case! This allows us to use the expressions which we have already derived for the linear case and gives us the variance of the parameters as usual,

$$\sigma_a^2(\hat{\mathbf{a}}) = \mathbf{C}(\hat{\mathbf{a}}) = \mathbf{H}^{-1}(\hat{\mathbf{a}}).$$

Similarly, the variance of the model function is now

$$\begin{aligned}\sigma_y^2(x,\hat{\mathbf{a}}) &= \sum_{j=1}^{m}\sum_{k=1}^{m} C_{jk}(\hat{\mathbf{a}}) d_j(x,\hat{\mathbf{a}}) d_k(x,\hat{\mathbf{a}}), \\ &= \mathbf{d}(x,\hat{\mathbf{a}})^T \mathbf{C}(\hat{\mathbf{a}}) \mathbf{d}(x,\hat{\mathbf{a}}), \qquad (9.35)\end{aligned}$$

which is formally the same as eq. 9.29 but with $\mathbf{x}$ replaced by $\mathbf{d}$. In other words, all we did was replace the basis functions, $F_j(x)$, by the derivatives $\partial y/\partial a_j$. Again, this translates exactly to eq. 9.29 if we consider a linear model function. Moreover, eq. 9.30 still holds

$$\frac{1}{n}\sum_{i=1}^{n}\frac{\sigma_y^2(x)}{\sigma_i^2} = \frac{1}{n}\sum_{j=1}^{m}\sum_{k=1}^{m} C_{jk}(\mathbf{C})^{-1}_{kj} = \frac{1}{n}\sum_{j=1}^{m} I_{jj} = \frac{m}{n}.$$

as it does in the case of constant errors,

$$\bar{\sigma_y^2} = \frac{1}{n}\sum_{i=1}^{n}\sigma_y^2(x_i) = \frac{m}{n}\sigma^2.$$

The only - but important - difference is that these results all depend on the solution parameters, $\hat{\mathbf{a}}$.

Let us now work through an example to illustrate how to use these newly found relations by considering a combination of an exponential background and a Gaussian peak form, i.e., the model function used for one of the individual peaks in Fig. 9.4.

$$y(x) = a\cdot\exp(-b\cdot(x-x_{\rm lo})) + c\cdot\exp\left(-\frac{(x-d)^2}{e^2}\right), \tag{9.36}$$

where $x_{\rm lo}$ is only introduced as a convenience and is no free parameter. We also need to prepare the derivatives with respect to the five parameters, a, b, c, d, and e:

$$\frac{\partial y}{\partial a} = \exp(-b\cdot(x-x_{\rm lo})) \tag{9.37}$$

$$\frac{\partial y}{\partial b} = -a\cdot x\cdot\exp(-b\cdot(x-x_{\rm lo})), \tag{9.38}$$

$$\frac{\partial y}{\partial c} = \exp\left(-\frac{(x-d)^2}{e^2}\right), \tag{9.39}$$

$$\frac{\partial y}{\partial d} = 2(x-d)c\exp\left(-\frac{(x-d)^2}{e^2}\right), \tag{9.40}$$

$$\frac{\partial y}{\partial e} = \frac{(x-d)^2}{e^3}c\exp\left(-\frac{(x-d)^2}{e^2}\right). \tag{9.41}$$

Note that we have removed the factor of 2 in the denominator of the exponent of the Gaussian to simplify matters. We'll need to account for that when we evaluate the fitted $e = \sqrt{2}\sigma$ of the Gaussian.

I present a few code snippet here to show how the fitting problem is solved.

```
def func(x,a,b,c,d,e):
    return a*exp(-b*(x-xlo)) + c*exp(-(x-d)**2/(e*e))
def dfda(x,popt):
    return exp(-popt[1]*(x-xlo))
def dfdb(x,popt):
    return - popt[0]*x*exp(-popt[1]*(x-xlo))
def dfdc(x,popt):
    return exp(-(x-popt[3])**2/(popt[4]*popt[4]))
def dfdd(x,popt):
    return 2*(x-popt[3])*popt[2}*exp(-(x-popt[3])**2/
           (popt[4]*popt[4]))
def dfde(x,popt):
    return ((x-popt[3])**2)/(popt[4]**3)*popt[2}*
           exp(-(x-popt[3])**2/(popt[4]*popt[4]))
```

The definition of the confidence and prediction bands is solved just as before,

```
def conf_band(x,yerr,popt,pcov,signif):
    n = len(x)
    np = len(popt)
    df = []
    df = append(df,dfda(x,popt))
    df = append(df,dfdb(x,popt))
    df = append(df,dfdc(x,popt))
    df = append(df,dfdd(x,popt))
    df = append(df,dfde(x,popt))
    df = reshape(df,(np,n))
    d = zeros(n)
    for j in linspace(0,np-1,np):
        for k in linspace(0,np-1,np):
            d += pcov[j,k]*df[j]*df[k]
```

..., and the fit is performed in the usual way.

```
def main():
    random.seed(12345678)
    n = 201
    x = linspace(xlo,xhi,n)
    y = func(x,a,b,c,d,e)
    ye = sqrt(y)
    y = y + random.normal(scale=ye)
    initial_values = array([1.5e10,0.9,1.5e6,102.,2.])
```

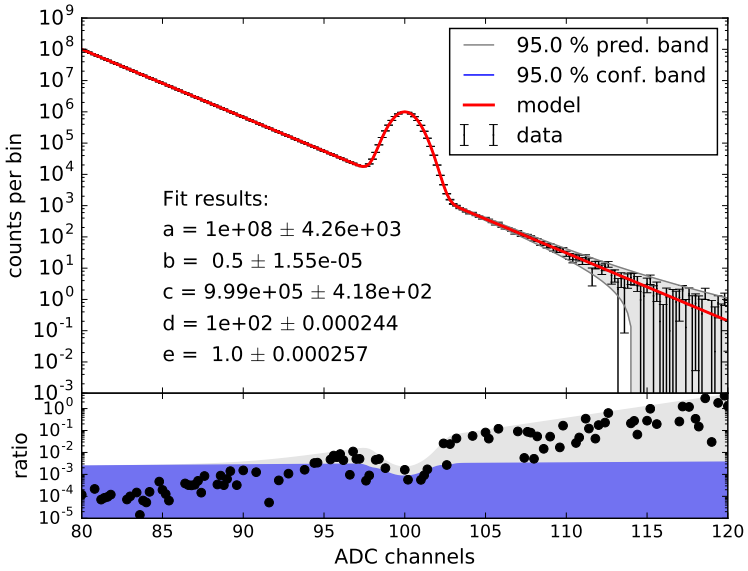


Figure 9.9: Fit of a non-linear model to non-linear data.

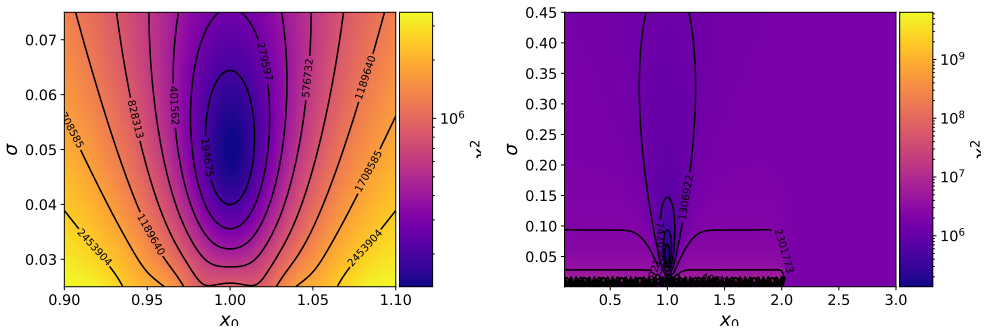
```
popt, pcov = curve_fit(func,x,y,p0=initial_values,
                       sigma=ye,absolute_sigma=True)
model_data = func(x,popt[0],popt[1],popt[2],popt[3],
                  popt[4])
```

..., etc.

The data were generated with the following parameters, $a = 1.e8$; $b = 0.5$; $c = 1.e6$; $d = 100.$; and $e = 1$. Moreover $x_{\rm lo} = 80$ was chosen. The data and fit are presented in Fig. 9.9. Because of the large dynamic range, the lower panel shows the ratio of $(y_i - y(x_i, \mathbf{a}))/y(x_i, \mathbf{a})$ instead of the usual residuals.

9.7 Non-Linear Minimization of χ^2: Levenberg-Marquardt and Trust-Region Methods

The aim of this section is to show you the principle how minimization algorithms for non-linear least-squares solvers work. Consider the Gaussian model function given in eq. 9.42. Using the methods presented in chapter 6 it is easy to generate a set of data, in this specific case I used standard normally distributed errors,



(a) Carefully chosen region around the global minimum.

(b) Much wider region around global minimum.

Figure 9.10: χ^2 "landscape" for fitting the Gaussian given in eq. 9.42 to Monte-Carlo-generated data. A contour plot is overlaid to improve visibility.

$\sigma = \mathcal{N}(0, 1)$ and generated 51 data points.

$$y(x) = \frac{100.}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(x - x_0)^2}{\sigma^2}\right) \tag{9.42}$$

We can now compute the χ^2 for this problem for different values of x_0 and σ and plot it as a heat map or contour plot. This is shown in Fig. 9.10 which illustrates the χ^2 "landscape" in which the minimizer must fund the *global* minimum. In the left-hand case, the landscape looks smooth and a minimizer will not have problems in finding the minimum. The right-hand panel shows the same χ^2 but for a much wider region around the global minimum and exhibits some weird "mountains" at low values for trial parameter σ and a plateau for trial parameter $2 \lesssim x_0$. If the minimizer starts at very small values for σ it may have difficulties getting out of the hilly region, if it starts on the plateau, it will take a while to find an appreciable slope for it to "know" in which direction to move towards the global minimum. This underlines the importance of finding good starting estimates for the parameters with which the minimizer can start its work.

The previous paragraph has already hinted how minimizers for non-linear problems work. Their key ingredient is to find the gradient of the χ^2 and follow it until you've reached the minimum of χ^2. There are various strategies to do this, `scipy` 's optimize function offers a number of methods which you can look up. I will not explain them all in detail here, but try to give you an idea how they work, following *Vanden-Berghen* (2004).

Consider a function, $F(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$, for which you want to find the minimum, $x_{\min} = \hat{x}$. Thus, F is a so-called "merit function", for instance the χ^2 of our problem. We expand F into a Taylor's series around some value x_0 and truncate

it at second order,

$$F(x_0+\delta) \approx F(x_0)+\left.\frac{\partial F}{\partial x_i}\right|_{x_0}\delta+\frac{1}{2}\delta^T\left.\left(\frac{\partial^2 F}{\partial x_i \partial x_j}\right)\right|_{x_0}\delta = F(x_0)+g(x_0)^T\delta+\frac{1}{2}\delta^T H(x_0)\delta. \tag{9.43}$$

The potentially large matrix which contains the partial second derivatives is the **Hessian**, H, of the function, F; $g(x) = \vec{\nabla}F$ is the gradient. When we're close to the minimum $\hat{x}$, the gradient will be close to zero and the Hessian will have positive curvature (otherwise we'd be at a local maximum). So when we're close to the minimum, we can solve for the minimum by solving

$$\vec{\nabla}F(\hat{x}_i+\delta_i) = g_i + H_i\delta_j = 0 \quad \Longleftrightarrow \quad H_i\delta_i = -g_i, \tag{9.44}$$

allowing us to take one step, δ_i to the minimum. This is similar to what we did when we were minimizing linear problems and is called a **Newton step** δ_i. Of course, we can also perform such steps when we're not exactly at the minimum and iterate our way towards the minimum. This is the idea behind Newton's algorithm or method:

1. choose a starting point, x_0
2. compute δ_i by solving $H_i\delta_i = -g_i$
3. set $x_{i+1} = x_i + \delta_i$
4. set $i = i + 1$ and continue with 2 unless $g_i \approx 0$ in which case you stop.

This is an example for a so-called "line-search" method in which one first determines the direction along which one attempts to find the solution. In these general line-search methods one often limits the length of the step by multiplying δ_i by some $0 < \alpha \leq 1$. Newton's method corresponds to $\alpha = 1$ and is very efficient when we are close to the solution, $\hat{x}$. Thus we should always use a full Newton step when we're close to the global minimum. Unfortunately, it can fail miserably when we're far from $\hat{x}$ because the gradient and Hessian can guide us in the wrong direction. But there is also a different problem which is much worse. What happens if the Hessian is not positive definite (as it is in the vicinity of the solution)? We then have negative curvature to the problem and the Hessian guides us far away from the current position. In other words, Newton's method will only converge if the Hessian, H, is positive definite. This can also be seen from the following considerations. In order to converge, the search direction must be a descent direction for the gradient, i.e.,

$$\delta^T g < 0. \tag{9.45}$$

Now we use g as determined by eq. 9.44 and insert it in eq. 9.45 to yield

$$-\delta_i^T H_i \delta_i < 0 \qquad \Longleftrightarrow \qquad \delta_i^T H_i \delta_i > 0, \tag{9.46}$$

in other words, the Hessian, H_i must be positive definite to ensure convergence of line-search methods. Only having a negative gradient will not be sufficient! So how can we ensure a positive definite Hessian which will allow us to solve the minimization problem?

The answer is as surprising as it is simple. Consider our approximation to $F(x+\delta)$ in eq. 9.43. It is just that, it is an approximation. Obviously, if we changed that approximation it would be a less good approximation, but not necessarily worse for finding a solution. So what if we used a different "Hessian" in eq. 9.43, one that certainly is positive definite? We define an approximation of the Hessian, D_i. For instance, we could approximate the Hessian using the identity matrix, I, so that $D_i = I$. In this case D_i is always positive definite and eq. 9.44 reduces to

$$\delta_i = -g_i. \tag{9.47}$$

It tells us to take a step exactly along the (negative) gradient. This is the steepest direction we can choose and this method is thus called the "steepest descent algorithm". While this can work, it is a very inefficient algorithm which can overshoot and wreak havoc on your optimization problem. So let us consider a slightly more complicated approximation to the Hessian, $D_i = H_i + \lambda I$. It retains some information of the Hessian, but we can choose λ large enough to ensure that D_i is positive definite. Repeating the Newton step (eq. 9.44) with this new approximation we get

$$D_i \delta_i = -g_i \qquad \Longleftrightarrow \qquad (H_i + \lambda I)\delta_i = -g_i. \tag{9.48}$$

This modified Newton step has two consequences. First, the Hessian, H_i, is negligible in the expression and we get a "steepest descent" estimate. But this steepest descent estimate for δ_i is no longer the full step size but a factor $1/\lambda$ smaller. This is the second effect. Note that we can now make the step size arbitrarily small by choosing λ appropriately large. This will also ensure that D_i is positive definite. So this algorithm will converge. *Vanden-Berghen* (2004) states that it can be proven that if one limits the step size $||\delta_i|| < \Delta_i$ then one maintains global convergence even if D_i is an indefinite matrix! This is the idea behind so-called "trust-region" algorithms which are implemented in `scipy` . Levenberg-Marquardt algorithms on the other hand, adapt the value of λ throughout the optimization, starting with a large λ and ending with a very small value for λ. If the optimization step was unsuccessful, increase the value of λ by a large factor (*Press et al.* (1989) choose a factor 10). If an optimization step was successful, λ is decreased by a large factor (and yes, *Press et al.* (1989) again choose a factor of 10). Once

you get to close to the minimum, your value of λ decreases a lot and you end up taking near perfect Newton steps. This is then where it pays off to retain the Hessian in the approximation, because now you can still use the information in it to compute the covariance matrix, i.e., the inverse of the Hessian, to estimate confidence limits on the parameters and fit function. The difficulty and problem with the Levenberg-Marquart algorithm lies in intermediate values for λ for which the new estimate (search direction) is determined by a mixture of steepest descent and Newton's step. This is a step in the direction defined by the perturbed "Hessian", B in which the perturbation, λI, has no geometrical meaning. It can point in any direction.

This difficulty in the Levenberg-Marquart algorithm is overcome in the more modern trust-region algorithms. They replace the adaptations of λ by a more controlled restriction to a "trust region" and uses Newton's method to search for a minimum of the quadratic approximation $Q(x_i)$ to the function $F(x_i)$ inside a generally circular trust region. This minimum then serves as the next starting point and one so iteratively searches for the global minimum. If the quadratic approximation is good, one expands the trust region, if it is bad, one tightens the trust region. Note that such trust-region methods require many evaluations of the function, F, and its approximation, Q. *Vanden-Berghen* (2004) gives an example for a simple trust-region algorithm. We consider a quadratic approximation, Q, of the function, F, to be minimized,

$$Q(x_0) = F(x_0) + g(x_0)^T \delta_0 + \frac{1}{2}\delta_0^T D(x_0)\delta_0,$$

where x_0 is the initial guess and $D(x_0)$ is the approximation to the Hessian also evaluated at x_0. The method then proceeds as follows:

1. Solve $D(x_0)\delta_0 = -g(x_0)$ subject to $||\delta_0|| < \Delta_0$. Typically, this would mean that $|\delta_0^T \delta_0| < \Delta^2$, i.e., $\Delta^2 < \delta^2$ where $|\Delta|$ is the radius of a circle around x_0.

2. Compute the quadratic approximation $Q(x)$ inside the trust region and use this to compute the degree of agreement, r_i between $F(x_i$ and $Q(x_i)$,

$$r_i = \frac{F(x_i) - F(x_i + \delta_i)}{Q(x_i) - Q(x_i + \delta_i)}$$

3. Update x_i and Δ_i according to the following table

$r_i < 0.01$ (bad iteration)	$0.01 < r_i < 0.9$ (good iteration)	$0.9 < r_i$ (very good iteration)
$x_{i+1} = x_i$	$x_{i+1} = x_i + \delta_i$	$x_{i+1} = x_i + \delta_i$
$\Delta_{i+1} = \frac{\Delta_i}{2}$	$\Delta_{i+1} = \Delta_i$	$\Delta_{i+1} = 2\Delta_i$

4. Increment $i = i + 1$ and go to step 1 unless $g_i \approx 0$ in which case you stop.

Note that this implies evaluation of g_i and D_i at each step, in other words, we need to compute derivatives or at least find a way to approximate them. This is done in the various implementations of trust-region algorithms.

Summary 9.4. Most non-linear minimization methods are of the Levenberg-Marquardt or Trust-Region type. The latter are more robust, the former tend to be faster if the objective function is smooth and well-behaved. Whichever method you choose, its success depends crucially on your choice of initial parameters. It is *always* worth while to put a lot of effort in finding good initial guesses for the parameters!

9.8 Orthogonal Distance Regression

So far, we have only considered problems in which the independent variable had no errors associated with it. This could be a calibration line with calibration standards as the independent variable, or a signal which is measured at fixed intervals in time. We considered the calibration of LND's detectors with the ^{207}Bi conversion electrons whose energies are know to high precision. On the other hand, in the introduction we mentioned the case of fitting a power law to measurements of energetic particles. Most space instruments can not afford to send back the full information about a measurement, and so the energy bins (as the independent variable) are often quite wide. How do we fit a model (e.g., a power law) to such data, in other words, how do we take into account the errors in the x axis as well as those in the y-axis? Other examples could be fitting an orbital curve to observations of a comet. These are measurements on the plane of the sky and in this case it is obvious that there are errors in the x and in the y direction.

Such situations are treated by **orthogonal distance regression** (ODR) algorithms or methods. Conceptually, all that needs to be done is to add the possibility of errors in the independent variable, x,

$$y_i = f(x_i, \mathbf{a}) + \varepsilon_i \quad \longrightarrow \quad y_i = f(x_i + \delta_i, \mathbf{a}) + \varepsilon_i,$$

where $\mathbf{a}$ is the vector containing all the parameters of the model f which describes the measurements, $y_i \in \mathbb{R}^1$, and the x_i can be multi-dimensional, i.e., $x_i \in \mathbb{R}^m$ where $m \geq 1$. An example would be the temperature of a plate which is heated non-uniformly. Then the temperature depends on both spatial dimensions, not only on one.

ODR considers the distance from the data point, (x_i, y_i), to the model curve, $f(x_i + \delta_i, \mathbf{a})$, and attempts to minimize the sum of squares, thus generalizing the

concept of ordinary least squares minimization. The square of the distance of the point (x_i, y_i) to the model is

$$r_i^2 = (f(x_i + \delta_i, \mathbf{a}) - y_i)^2 + \delta_i^T \delta_i,$$

and follows from the Pythagorean theorem for the radius of the smallest possible circle around (x_i, y_i) which is tangent to the model curve, f. This expression does not yet account for the possibility that measurements in x and y can have different precisions, and so we generalize the previous expression to the weighted orthogonal distance,

$$\tilde{r}_i^2 = (f(x_i + \delta_i, \mathbf{a}) - y_i)^2 + \delta_i^T d_i^2 \delta_i, \tag{9.49}$$

where the $d_i \in \mathbb{R}^{m \times m}$ are positive definite diagonal m by m matrices which weight the individual components δ_i. The "best" set of parameters or the solution of this optimization problem is then found by minimizing the sum of squares of $\tilde{r}_i$,

$$\min_{\mathbf{a},\delta} \left(\sum_{i=1}^{n} w_i^2 \left(f(x_i + \delta_i, \mathbf{a}) - y_i \right)^2 + \delta_i^T d_i^2 \delta_i \right), \tag{9.50}$$

where the weights, $w_i > 0$, give the weights of the individual data points. Thus eq. 9.50 defines the the orthogonal distance regression problem that needs to be solved. *Boggs et al.* (1987) and *Boggs* and *Rogers* (1989) show that this problem can be expressed as a non-linear least squares problem (see sec. 9.6) with $n + nm$ observations or measurement points and $p + nm$ unknowns. They treat the unknowns as a vector, $\eta^T = (\mathbf{a}^T, \delta_1^T, \delta_2^T, \ldots, \delta_n^T)$. With this, they express the sum of squares in eq. 9.50 as

$$S(\eta) \doteq G(\eta)^T \Omega G(\eta).$$

Here $G(\eta)$ is a vector valued function in which the ith element is given by

$$g_i(\eta) = \begin{cases} f(x_i + \delta_i, \mathbf{a}) - y_i & i = i, \ldots, n, \\ \eta_{p+i-n} & i = n+1, \ldots, n+nm \end{cases}$$

The matrix $\Omega \in \mathbb{R}^{(n+nm)\times(n+nm)}$ is diagonal and represents the weights w and d. In other words,

$$\Omega = \begin{bmatrix} W & \\ & D \end{bmatrix}, \tag{9.51}$$

where $W \in \mathbb{R}^{(n \times n)}$ is the diagonal matrix with (diagonal) entries w_i^2 and $D \in \mathbb{R}^{(nm \times nm)}$ is the diagonal matrix which consists of diagonal sub-matrices $w_i^2 d_i^2$ (remember that $x_i \in \mathbb{R}^m$). Because of the diagonality of Ω, the least squares condition of eq. 9.50 can be written as

$$\min_\eta S(\eta) = \min_\eta \sum_{i=1}^{n+nm} \Omega_{ii} g_i^2(\eta), \tag{9.52}$$

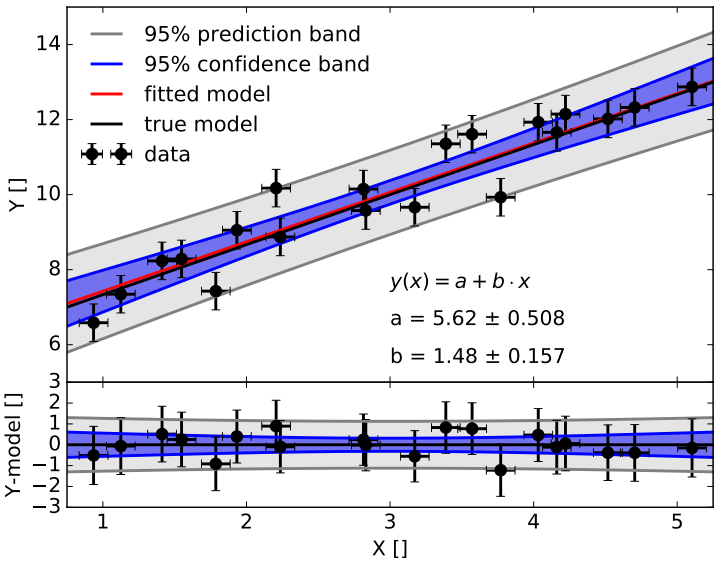


Figure 9.11: Example for fitting with ODR.

where, of course, Ω_{ii} are the diagonal (ii-th) elements of Ω.

Boggs et al. (1987) solve this problem using a trust-region Levenberg-Marquardt algorithm which is made available in `scipy` as the `scipy .odr` package. You can find how to use it at `https://docs.scipy.org/doc/scipy/reference/odr.html`.

Example 9.3. Figure 9.11 shows an example of data fitted with `scipy` 's `odr` package. A linear function

$$y(x) = 5.5 + 1.5 \cdot x$$

was used as input. To generate the "data" points, I randomized them with $\sigma_x = 0.05$ and $\sigma_y = 0.5$, where I only randomized the x values after I had randomized the y values. The following code snippets give you an idea how to use `odr`. The data were generated as follows:

```
def main():
    random.seed(145)
    a = 5.5; b = 1.5  #intercept and slope
    pmod = [a,b]
    n = 20            #number of data points
    signif = 0.05  #significance level
    sx = 0.1; sy = 0.5        #errors in x and y
    xlo = 1.; xhi=5. #boundaries
```

```
x = linspace(xlo,xhi,n)
xp = linspace(xlo-0.25,xhi+0.25,n) #for plots only
ym = lin(pmod,x)  #model
y = lin(pmod,x)+random.normal(0.,sy,n) #data y
xd = x + random.normal(0.,sx,n)        #data x
```

To fit a linear model to the data you need the following:

```
def lin(B,x):
    return B[0] + B[1]*x

def lin_derivs(B,x):
    return [ones(len(x)),x]

    #in main() add these lines
def main():
    linear = odr.Model(lin)
    mydata = odr.Data(x, y, wd=1./sx*ones(len(x)),
                      we=1./sy*ones(len(y)))
    myodr = odr.ODR(mydata, linear, beta0=[1.3, 2.5])
    myoutput = myodr.run()
    myoutput.pprint()
    popt = myoutput.beta        #fitted parameters
    perr = myoutput.sd_beta     #errors of parameters
    pcov = myoutput.cov_beta    #covariance matrix
    pdel = myoutput.delta       #Array of estimated errors
            #in input variables, of same shape as 'x'
    peps = myoutput.eps         #Array of estimated errors in
            #response variables, of same shape as 'y'
    paras = ['a','b','c','d','e','f']
    print("fitted parameters are:")
    for i in range(0,len(popt)):
        print(paras[i], '=', popt[i], '+/-', perr[i])
    #get the derivatives for confidence bands
    dfdp = lin_derivs(popt,x)
    #get the confidence bands
    cbu, cbl = conf_band(lin,dfdp,x,y,sy,signif,popt,
                         pcov,abs_weights=False)
```

where `conf_band` is defined on page 161.

```
def conf_band(func,dfdp,x,y,yerr,signif,popt,pcov,
              abs_weights=False):
    np = len(popt) #number of parameters
    n = len(x)     #number of data points
    alpha = 1.-signif/2
    tval = t.ppf(alpha,n-np) #student's t
    r_chisq, p = chisquare(y,func(popt,x),np)
    if abs_weights:
        cov_scale = 1.
    else:
        cov_scale = r_chisq
    d = zeros(n)
    for j in range(np):
        for k in range(np):
            d += dfdp[j]*dfdp[k]*pcov[j,k]
    d1 = sqrt(yerr**2 + d)
    d2 = sqrt(cov_scale*d)
    ci1 = tval*d1 #prediction band for a new measurement
    ci2 = tval*d2 #confidence band for the model
    return ci1,ci2
```

9.9 Maximum Likelihood Fitting Techniques

Consider measurements $z_i(x_i)$ of some "true" value $Z(x_i)$. They will be distributed randomly around the true value according to some probability density function $p(z, Z)$. This function is very often *implicitly assumed* to be a Gaussian or normal distribution. Of course, this is not necessarily true. The true underlying probability density of the errors, $p(z, Z)$ ought to be considered when fitting a model to the data. Much too often, this is not done. Let us define the negative logarithm of the probability density function, $p(z, Z)$,

$$\rho(z, Z) \doteq -\ln\left(p(z, Z)\right).$$

Apart from the measurements, we (hopefully) have a model for the physics underlying the result, i.e., a model for the z_i, for instance, a linear model, $Z \doteq \mathbf{A}\vec{f}$, where $\vec{f} = \vec{f}(\vec{x})$ could, for instance, be a model energy spectrum. The matrix, $\mathbf{A}$, may depend on M parameters $\vec{a} = (a_1, \ldots, a_M)$ and can also be a function of energy, $\vec{x}$. It could contain information about the geometry factor of the instrument as well as about the detection efficiency (see chapter 2 for a discussion of the

geometry factor and detection efficiencies.). In this case, $\vec{f}$ would be the expected number of counts in energy bins $\vec{x}$. Thus the expected measurements $\vec{z}$ can be written as a function $Z(x_i, \vec{a})$. In some cases, the only free parameters are $\vec{f}$, because the model function is completely determined by physics and physics-based simulations. This is the case for the detection of energetic particles with many of our particle instruments.

With such a model, $Z(x_i, \vec{a})$, we can determine the probability, P, that the n measurements, z_i, would occur for a given set of parameters $\vec{a}$. We rewrite it using $\rho(z, Z)$ defined above.

$$P = \prod_{i=1}^{n} p(z_i, Z(x_i, \vec{a}))\Delta z = \prod_{i=1}^{n} \left\{ e^{-\rho(z_i, Z(x_i, \vec{a}))} \ \Delta z \right\},$$

where the Δz is needed to make this meaningful. While it may be possible to define P with $\Delta z = 1$ for a Poisson distributiuon, in which case the z_i are natural numbers, it is not meaningful for other cases in which z_i are real numbers. A good model will result in a large probability, P. Thus, we want to adjust the parameters $\vec{a}$ such as to maximize the probability, P, i.e., maximizing the likelihood of the model. This, of course, is where the name "Maximum Likelihood" comes from, see Sec. 7.5. Following that logic, to maximize P, its logarithm,

$$\sum_{i=1}^{n} \ \rho(z_i, Z(x_i, \vec{a})) - n \ln \Delta z,$$

must be minimized. The term $-n \ln \Delta z$ is constant and therefore does not contribute to the minimization. It can be discarded.

To begin this treatment, we consider for the time being a $P(z, Z)$ which is a normal distribution. Then

$$\sum_{i}^{n} \rho(z_i, Z(x_i, \vec{a})) = \sum_{i}^{n} \frac{(z_i - Z(x_i, \vec{a}))^2}{2\sigma_i^2}.$$

This quantity needs to be minimized. Obviously, this is the case when the quadratic mean of the distances $z_i - Z$ is as small as possible. This is exactly the meaning and origin of the "least squares" method. In other words, the least-squares method is a maximum likelihood method if and only if the measurement errors are normally distributed.

> Least-squares fitting techniques are maximum likelihood methods if and only if the probability density functions which describe the distribution of data points around the model is a normal distribution!
> Once you have successfully fitted a model to your data, you should verify that the residuals are normally distributed around zero!

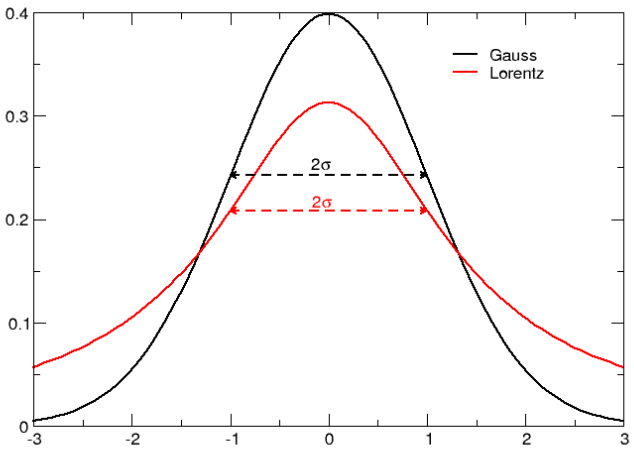


Figure 9.12: Illustration of the normal and Lorentzian distributions.

To illustrate this, let us try this out with a linear function (straight line). We minimize

$$\chi^2 \doteq \sum_{i=1}^{n} \left(\frac{z_i \; - \; z(x_i; a_1, \ldots, a_M)}{\sigma_i} \right)^2 .$$

This method is often called the χ-squared method because the probability density function of χ^2 is distributed as χ^2 with $n-M$ degrees of freedom if the measurement errors are again normally distributed.
To minimize χ^2, we take the partial derivatives of χ^2 with respect to a_i, equate to zero, and obtain a system of equations. If the model z is linear this is also linear and can easily be solved for the parameters a_i,

$$0 = \sum_{i=1}^{n} \left(\frac{z_i - z(x_i, \vec{a})}{\sigma_i^2} \right) \left(\frac{\partial z(x_i; \ldots, a_k, \ldots)}{\partial a_k} \right) \quad \text{for } k = 1, \ldots, M.$$

It is this property that makes linear models so easy to fit. On the other hand, if the model is not linear in a_k, some method needs to be used which can deal with a non-linear system of equations. This can be done for instance with the algorithm by Levenberg and Marquardt (e.g. *Press et al.*, 1989).

9.9.1 What happens with other distributions, $p(z)$?

Often, measurement errors are not normally distributed but follow some other distribution, p. Figure 9.12 shows a normal distribution (black) and a Lorentz-

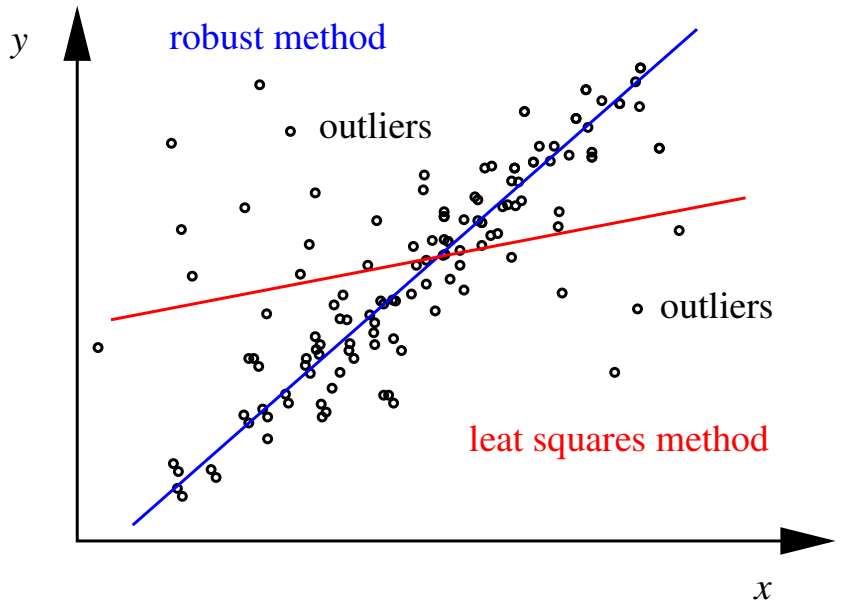


Figure 9.13: Outliers can ruin a nice correlation unless you use robust fitting techniques.

distribution (red). Both have the same area in the interval $-3 < x < 3$. Obviously, the Gaussian distribution is "narrower" and the Lorentz distribution has larger "tails". It is these tails that would cause the least squares method to fail.

9.9.2 Robust methods

The outliers of a Lorentz distribution will be given a larger weight than they "deserve" by the least-squares method. The normal distribution underlying "least squares" considers these outliers very unlikely even though they aren't because the true underlying distribution is Lorentzian. A more robust estimation method could be obtained if we used the true underlying measurement statistics instead of the wrong one.

The probability P that our model $z(x_i, \vec{a})$ could realize the n measurements $z_i(x_i)$ is still given by

$$P = \prod_{i=1}^{n} p_i = \prod_{i=1}^{n} \{e^{-\rho(z_i, z(x_i,\vec{a}))}\ \Delta z\}$$

Such an exponential form can always be achieved. To maximize the probability P

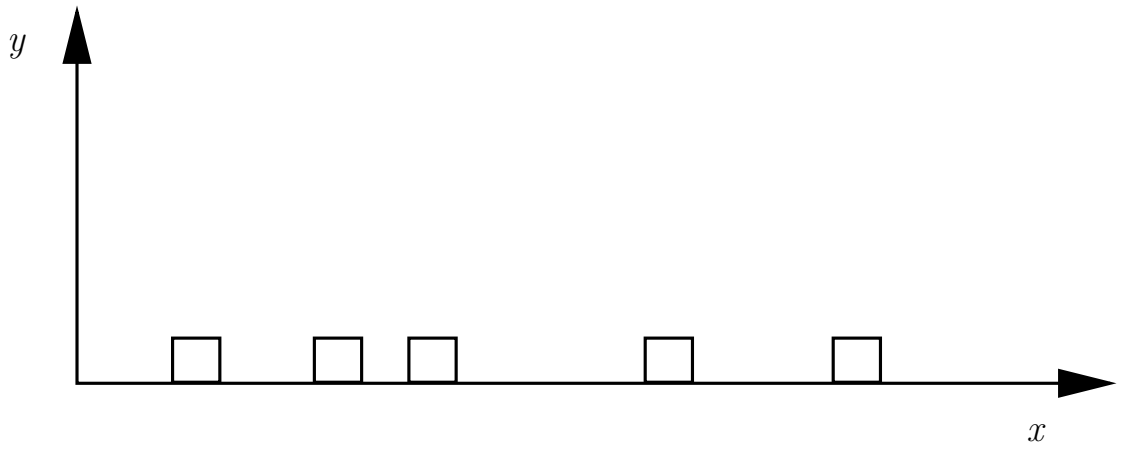


Figure 9.14: Illustration of discrete counts which need to be fitted with a model function.

we can also minimize the following expression.

$$\sum_{i=1}^{n} \rho(z_i, z(x_i, \vec{a})) = -\sum_{i=1}^{n} \ln(p_i) \tag{9.53}$$

Note that using the exponential formulation allows us to transform the product into a sum. It is also worth noting that if $\rho(z_i, z(x_i, \vec{a})$ is not Gaussian, but more complicated, e.g., a Lorentzian, then we will necessarily end up with a set of non-linear equations when trying to minimize eq.9.53.

9.9.3 An Example: Fitting a straight line to rare events

Consider an experiment in which events are classified into classes or bins (in space, time, energy, etc.). If the events are rare, then the subjective choice of intervals (bins or bin widths) plays an important role in the analysis of the data. On the one hand, the bins need to be chosen large (wide) enough to contain a statistically significant number of events. On the other hand, this "smears out" the originally available information.

The solution of this dilemma lies in careful consideration of the underlying random process. In every bin there is an expectation value, $\langle z_i(x_i)\rangle$, for the corresponding measurement. This needs to be modeled as accurately as possible. The model should, on average, equal the expectation value,

$$\langle z_i(x_i)\rangle = z(x_i, \vec{a}).$$

If the events are rare, then one of the possible probability distributions is the

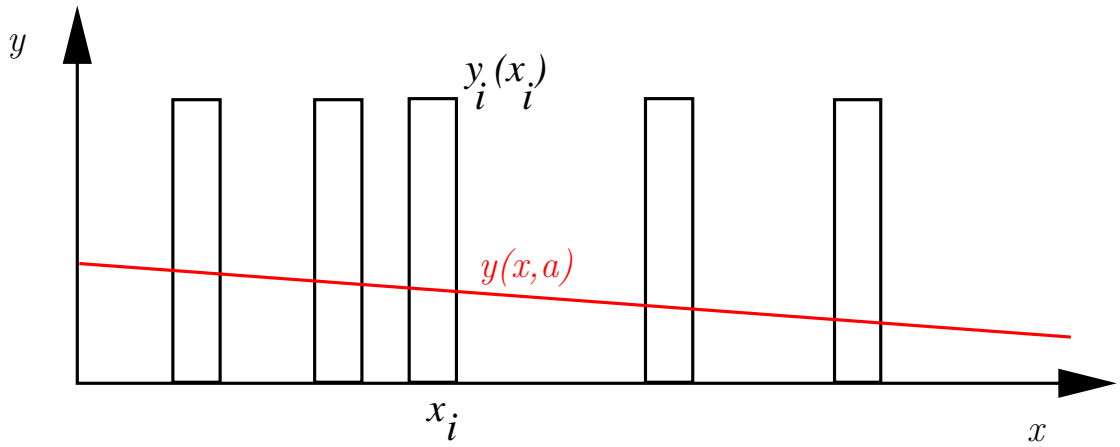


Figure 9.15: Illustration of how discrete counts can be fitted with a model function for the expectation value.

Poisson distribution

$$P(z_i) = \frac{z(x_i, \vec{a})^{z_i} \cdot e^{-z(x_i,\vec{a})}}{z_i!}.$$

9.9.4 Poisson-distributed Measurements

Consider the Poisson distribution

$$p(z_i) = \frac{\lambda_i^{z_i}\ e^{-\lambda_i}}{z_i!},$$

where λ_i is the expectation value $\lambda_i = \langle z_i(x_i) \rangle$. Now take the logarithm

$$\rho_i = -\ln p(z_i) = -\left[z_i \ln \lambda_i - \lambda_i - \ln(z_i!)\right].$$

The sum, $\sum_i \rho_i$, needs to be minimized to maximize the probability, $\prod p_i$, for the model (or expectation value), $\lambda(\vec{a}, \vec{x})$, to fit the data, $\vec{z}$. Here the $\vec{x}$ are the independent variable (e.g., energy or temporal bins), and the $\vec{a}$ are the free parameters

which need to be fitted. It is now straightforward to obtain the set of equations

$$\begin{aligned}
\frac{\partial \rho}{\partial a_1} &= \sum_{i=1}^{n} \frac{\partial \lambda(x_i, \vec{a})}{\partial a_1} - z_i \frac{\partial \ln(\lambda(x_i, \vec{a}))}{\partial a_1} = 0, \\
\vdots &= \vdots \\
\frac{\partial \rho}{\partial a_k} &= \sum_{i=1}^{n} \frac{\partial \lambda(x_i, \vec{a})}{\partial a_k} - z_i \frac{\partial \ln(\lambda(x_i, \vec{a}))}{\partial a_k} = 0, \\
\vdots &= \vdots \\
\frac{\partial \rho}{\partial a_M} &= \sum_{i=1}^{n} \frac{\partial \lambda(x_i, \vec{a})}{\partial a_M} - z_i \frac{\partial \ln(\lambda(x_i, \vec{a}))}{\partial a_M} 00,
\end{aligned}$$

which is no longer linear. Nevertheless, it can often still be solved.

9.9.5 A Worked Example: Fitting a Power-Law to a Particle Energy Spectrum

Because of the underlying physical processes, energy spectra of energetic particles in space are often well described by (piece-wise) power laws in energy. Because of limitations in the data volume that can be transmitted back to Earth, the data have to be binned into a limited number of (often coarse) energy bins. The differential flux, $\mathrm{d}J/\mathrm{d}E$, can be modeled by

$$\frac{\mathrm{d}J}{\mathrm{d}E} = J_0 \cdot E^{-\gamma} \tag{9.54}$$

which has two free parameters, J_0 and γ. The measurements, however, are given by counts, z_i, in discrete energy bins, $E_i \pm \frac{\Delta E_i}{2}$, which do not necessarily need to be evenly distributed. If we are willing to wait long enough, we will have lots of counts in every energy bin and can use the usual least-squares fitting techniques. If we don't have enough counts to do so, we could also combine bins into a smaller number of bins at the cost of loosing information about the particle's energy. On the other hand, if we want to obtain higher time and energy resolution, we need to account for the non-normal (i.e., Poisson) distribution of counts in the different energy bins.

Note that the physical model for the spectrum which is given by eq. 9.54 is not the model for the measurements. We first need to translate this physical model into a measurement model. The expected number of counts , λ_i, in a given energy bin, E_i, will be given by

$$\lambda(J_0, \gamma, E_i, \Delta E_i) = J_0 \cdot E_i^{-\gamma} \Delta E_i. \tag{9.55}$$

As expected, the expectation value, λ, is proportional to the bin width, ΔE_i, and intensity, J_0. The bin width is a known quantity and so does not need to be fitted. To obtain the model parameters J_0 and γ which maximize the likelihood for the model, λ, to describe the measurments, z_i, we now need to evaluate

$$\begin{aligned}
\frac{\partial \rho}{\partial J_0} &= \sum_{i=1}^{n} \left(\frac{\partial \lambda(J_0, \gamma, E_i, \Delta E_i)}{\partial J_0} - z_i \frac{\partial \ln(\lambda(J_0, \gamma, E_i, \Delta E_i))}{\partial J_0} \right) = 0, \\
&= \sum_{i=1}^{n} E_i^{-\gamma} \Delta E_i - z_i \frac{1}{\lambda_i} E_i^{-\gamma} \Delta E_i = 0, \\
&= \sum_{i=1}^{n} E_i^{-\gamma} \Delta E_i \left(1 - \frac{z_i}{\lambda_i(J_0, \gamma, E_i, \Delta E_i)} \right) = 0, \\
&= \sum_{i=1}^{n} \left(E_i^{-\gamma} \Delta E_i - \frac{z_i}{J_0} \right) = 0, \qquad (9.56)
\end{aligned}$$

and, similarly,

$$\begin{aligned}
\frac{\partial \rho}{\partial \gamma} &= \sum_{i=1}^{n} \left(\frac{\partial \lambda(J_0, \gamma, E_i, \Delta E_i)}{\partial \gamma} - z_i \frac{\partial \ln(\lambda(J_0, \gamma, E_i, \Delta E_i))}{\partial \gamma} \right) = 0, \\
&= \sum_{i=1}^{n} \left(-J_0 \Delta E_i \ln(E_i) E_i^{-\gamma} + z_i \ln(E_i) \right) = 0. \qquad (9.57)
\end{aligned}$$

Note that the bin width, ΔE_i, must be included even if it does not appear in the original physical model, eq. 9.54 because the model of the measurement must include the measurement process. The maximum-likelihood fit is now given by the solution (J_0, γ) of equations 9.56 and 9.57. You can easily see that this system of equations for (J_0, γ) is non-linear and that it depends on the bin width, ΔE_i as a parameter, and, of course, on the measurements, z_i.

Example 9.4. Radon is one of the major health hazards due to radioactivity. Its isotope ^{222}Rn has a half life of 3.8 days and α-decays into ^{218}Po which in turn α-decays with a half life of 3.1 minutes. We can model the α-decays of ^{218}Po using the following code snippet:

```
from numpy.random import exponential
half_life = 3.1 #half life of 218Po in minutes
scale = half_life/log(2.) #so we can use exp(-time/scale)
size = 10       #number of decay time differences
rn = exponential(scale,size) #generate random times
```

Note that we're only modelling 11 decays or 10 time differences because we want to use a maximum-likelihood fit to the data. You can check that this does indeed result an exponential distribution by increasing `size` to a larger value. We define the merit function as follows:

```
def merit_func(par, *args):
    x = args[0]  #the x values (i.e., time differences)
    y = args[2]  #the data
    xw = args[1] #width of the x bins
    merit = 0.
    par = abs(par)
    for i in xrange(0,len(par),2):
        merit += abs(sum(par[i]*exp(-x/abs(par[i+1]))*xw)\
            - sum(y*log(par[i]*exp(-x/abs(par[i+1]))*xw)))
    print 'merit = ', merit
    if par[0] < 1.e-10 or par[1] < 0.
                or isnan(merit): #unrealistic values
        print 'unrealistic: merit = {}'.format(merit)
        for i in xrange(0,len(par),2):
            print i, 'par[{}] = {}, par[{}] = {}'
                    .format(i,par[i],i+1,par[i+1])
            merit = par[0]*1.e22 + par[1]*1.e+19
        return merit
    return merit
```

It is called by the following lines

```
  def fit_dt_dist(xdat, xw, ydat, par):
    "prepare for and fit an exponential to the data"
    args = (xdat, xw, ydat)
    [popt, fopt, gopt, Bopt, func_calls, grad_calls, warnflg] = \
            fmin_bfgs(merit_func, x0=abs(par), args=args, \
            full_output=True)
    return popt, Bopt

true_model = size/scale*exp(-plot_bins/scale)
esti_scale = sum(rn)/size
esti_model = size/esti_scale*exp(-plot_bins/esti_scale)
par = array([size/esti_scale,esti_scale])
popt, Bopt = fit_dt_dist(plot_bins, dbins, hist, par)
fitted_model = popt[0]*exp(-plot_bins/popt[1])
```

...and we plot the results with

```
  fig , ax = subplots()
ax.set_xlabel('time between decays [minutes]',fontsize=14)
ax.set_ylabel(r'counts per bin [minute$^{-1}$]',fontsize=14)
ax.plot(plot_bins,fitted_model,color='k',lw=3,\
        label='fitted model')
plt.bar(plot_bins-0.5*0.75*dbins,hist/dbins,label='data',\
        width=0.75*dbins,align='edge',alpha=0.5)
ax.plot(plot_bins,true_model,color='blue',lw=2,
        label='true model')
ax.plot(plot_bins,esti_model,color='red',lw=2,ls='dashed',
        label='estimated model/\ninitial guess')
ax.errorbar(plot_bins,hist/dbins,sqrt(hist),fmt='ko')
ax.legend()
ax.tick_params(axis='both', which='major', labelsize=14)
ax.set_ylim(bottom=0)
savefig('exp_decay_maxli_fit.pdf')
show()
```

All of this results in the plot shown in fig. 9.16 (or a similar one, depending on the random numbers produced by the `exponential()` random number generator). Note that we have allowed for uneven bins in this example. The blue curve shows the true, underlying distribution, from which the data shown in the histogram have been drawn. The initial guess was determined as shown in eq. 9.58

$$\tau = \frac{1}{n}\sum_i^n \tau_i \quad \text{and} \quad \text{model estimate} = \frac{n}{\tau} e^{-t/\tau}, \tag{9.58}$$

as is also shown in the code snippets on page 169.

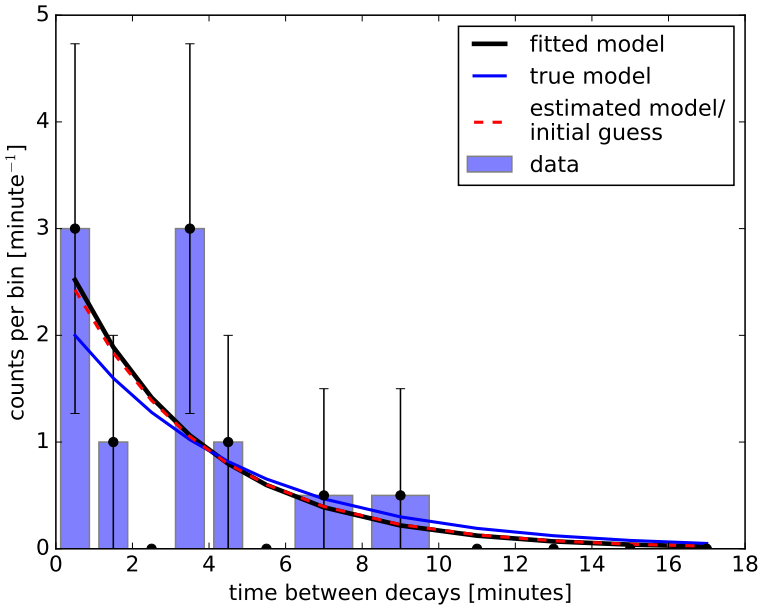


Figure 9.16: A set of 10 random waiting times between decays of ^{218}Po which has a half life of 3.1 minutes. The histogram shows the "data", counts binned into minute wide bins for small values and wider 2-minute bins for larger times. The blue curve shows the true, underlying distribution, the dashed red line the estimate explained in the text, and the black curve the fitted model.

Chapter 10

Detecting Breakpoints in a Data Series

10.0.1 Standard Normal Homogeneity Test (SNHT)

Quite often one needs to find discontinuities in data, also called breakpoints. Figure 10.1 shows a simulated example for which I generated artificial data which are distributed as a Poissonian distribution around expectation values $\lambda_1 = 4.5$ and $\lambda_2 = 6.5$. In other words, they mimick count rates with low counting statistics.

Consider the situation sketched in Fig. 10.1. We have data in the form of a time series and want to find the breakpoint. To do so we perform the following steps:

1) If needed, homogenize the data: If you have wildly different data from different sources with different properties such as variances, it makes sense to homogenize these data sets by computing a standardised data set

$$Z_i \doteq (q_i - \bar{q})/\sigma_q, \tag{10.1}$$

where q_i are the data points of data series Q, σ_q their standard deviation, and Z_i the standardized series with zero mean and standard deviation $\sigma_z = 1$.

2) Define the null and alternate hypotheses for a discontinuity in the data:

$$H_0: \quad z_i \sim N(0,1) \quad \text{for} \quad i = 1, 2, \ldots, n, \tag{10.2}$$

$$H_1: \quad \begin{cases} z_i \sim N(\mu_1, 1) & \text{for} i = 1, \ldots, a \\ z_i \sim N(\mu_2, 1) & \text{for} i = a+1, \ldots, n \end{cases}, \tag{10.3}$$

where $N(\mu, \sigma)$ as usual denotes the normal distribution with mean μ and standard deviation σ.

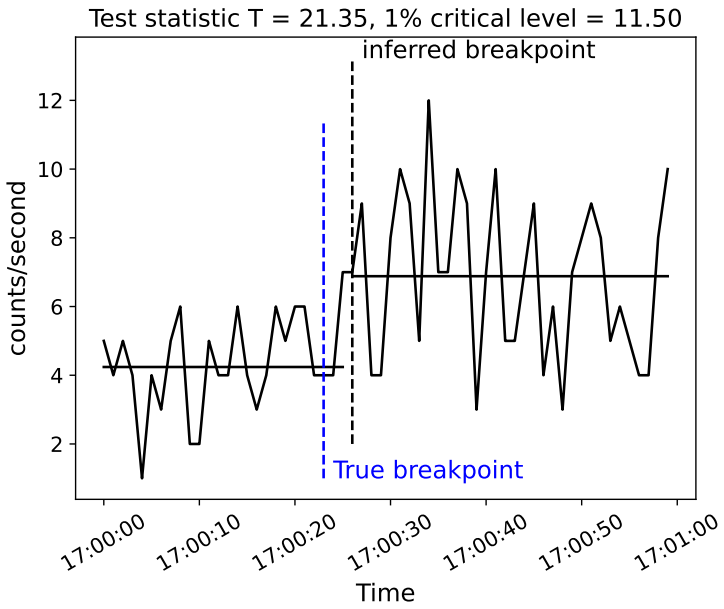


Figure 10.1: Simulated data with a breakpoint (discontinuity) around 17:00:24. The vertical blue dashed line indicates the true breakpoint whereas the vertical black dashed line shows the inferred breakpoint. Horizontal black lines show the averages of the data to the left and right of the breakpoint.

3) Test the validity of H_0 against H_1. To do so, *Andersson* (1986) derived a test statistic, T, which relies on the probabilities of the two hypotheses based on the observations z_i. He defined

$$T \doteq \max(a\bar{z}_1^2 + (n-a)\bar{z}_2^2), \text{for} 1 \leq a \leq n-1,) \tag{10.4}$$

where $\bar{z}_1$ and $\bar{z}_2$ are the averages of z_i before and after potential "breakpoints" located at a. The point a which maximizes the quantity inside the parenthesis in eq. 10.4 is then the most likely breakpoint.

4) We reject the null hypothesis, H_0, in favor the the alternate hypothesis, H_1 if the test statistic, T, is larger than a critical value for a certain significance level. The most complete table of critical values which I found is given in *Khaliq* and *Ouarda* (2007). Some of these values are given in Tab. 10.1.

This test is called the standard normal homogeneity test (SNHT). The result of such a test with simulated values is shown in Fig. 10.1. In that case, the test statistic, T, exceeded the critical level and we can believe the result at the 1% confidence level.

Sample	crit. level (%)	
size	95	99
10	5.637	6.769
20	7.089	9.113
30	7.747	10.153
40	8.151	10.771
50	8.432	11.193
70	8.814	11.737
100	9.167	12.228
150	9.519	12.694
200	9.741	12.982
400	10.202	13.542
800	10.580	13.975
1000	10.692	14.105

Table 10.1: Table of some critical values of the SNHT test statistic T for different sample sizes and the 95% and 99% critical (confidence) levels. Values were taken from *Khaliq* and *Ouarda* (2007).

Bibliography

Andersson, H.: "A homogeneity test applied to precipitation data". *J. Climatology*, **6**, (1986), 661 – 675.

Anscombe, F. J.: "Graphs in statistical analysis". *The American Statistician*, **27**(1), (1973), 17–21. ISSN 00031305.

Bevington, P. R. and D. K. Robinson: *Data reduction and error analysis for the physical sciences; 3rd ed.* McGraw-Hill, New York, NY (2003).

Boggs, P. T., R. H. Byrd, and R. B. Schnabel: "A Stable and Efficient Algorithm for Nonlinear Orthogonal Distance Regression". *SIAM Journal on Scientific and Statistical Computing*, **8(6)**, (1987), 1052–1078.

Boggs, P. T. and J. E. Rogers: "The Computation and Use of the Asymptotic Covariance Matrix for Measurement Error Models". Technical Report NISTIR 89–4102, U.S. Departmen of Commerce, National Institute of Standards and Technology (1989). Revised 1990.

Brink, D.: *Essential of Statistics*. David Brink & Ventus Publishing ApS (2010). ISBN 978-87-7681-408-3.

Chambers, J. M., W. S. Cleveland, B. Kleiner, and P. A. Tukey: *Graphical Methods for Data Analysis*. Wadsworth (1983).

Filliben, J. J.: "The Probability Plot Correlation Coefficient Test for Normality". *Technometrics*, 111 – 117.

Galvin, A. B., L. M. Kistler, M. A. Popecki, C. J. Farrugia, K. D. C. Simunac, L. Ellis, E. Möbius, M. A. Lee, M. Boehm, J. Carroll, *et al.*: "The Plasma and Suprathermal Ion Composition (PLASTIC) Investigation on the STEREO Observatories". *Space Sci. Rev.*, **136**, (2008), 437–486.

Gloeckler, G., J. Cain, F. M. Ipavich, E. O. Tums, P. Bedini, L. A. Fisk, T. Zurbuchen, , P. Bochsler, J. Fischer, *et al.*: "Investigation of the composition of

solar and interstellar matter using solar wind and pickup ion measurements with SWICS and SWIMS on the ACE spacecraft". *Space Sci. Rev.*, **86**, (1998), 497 – 539.

Gloeckler, G., J. Geiss, H. Balsiger, P. Bedini, J. C. Cain, J. Fisher, L. A. Fisk, A. B. Galvin, F. Gliem, and D. C. Hamilton: "The Solar Wind Ion Composition Spectrometer". *Astron. Astrophys. Suppl.*, **92**, (1992), 267–289.

Khaliq, M. N. and T. B. M. J. Ouarda: "Short Communication On the critical values of the standard normal homogeneity test (SNHT)". *Int. J. Climatol.*, **27**, (2007), 681 – 687.

Knappmann, A. C.: *Test von Festkörperdetektoren in Vorbereitung für LND*. Bachelor thesis, Christian-Albrechts-Universität zu Kiel, Kiel, Germany (2016).

Landau, L.: "On the energy loss of fast particles by ionization". *Journ. Phys. USSR*, **8.4**, (1944), 201 – 205.

Owen, C. J., R. Bruno, S. Livi, P. Louarn, K. Al Janabi, F. Allegrini, C. Amoros, R. Baruah, A. Barthe, M. Berthomier, *et al.*: "The Solar Orbiter Solar Wind Analyser (SWA) suite". *Astron.& Astrophys.*, **642**, (2020), A16.

Patrignani, C. *et al.*: "Review of Particle Physics". *Chin. Phys.*, **C40**(10), (2016), 100001.

Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling: *Numerical Recipes: The Art of Scientific Computing*. Cambridge Univ. Press, New York (1989).

Richter, P. H.: "Estimating Errors in Least-Squares Fitting". *TDA Progress Report 42-122*, 1–31.

Stone, E. C., A. M. Frandsen, R. A. Mewaldt, E. R. Christian, D. Margolies, J. F. Ormes, and F. Snow: "The Advanced Composition Explorer". *Space Sci. Rev.*, **86**, (1998), 1–22.

Sullivan, J. D.: "Geometrical factor and directional response of single and multi-element particle telescopes". *Nucl. Instrum. Meth.*, **95**, (1971), 5 – 11.

Sullivan, J. D.: "Errata: Geometrical factor and directional response of single and multi-element particle telescopes". *Nucl. Instrum. Meth.*, **98**, (1972), 187.

Vanden-Berghen, F.: *Constrained, non-linear, derivative-free, parallel optimization of continuous, high computing load, noisy objective functions*. Ph.D. thesis, Université Libre de Bruxelles, Faculté des Sciences Appliquées, Service: IRIDIA (2004).