MATLAB Statistical Toolbox

Radu T. Trîmbiţaş

Contents

Chapter 1

Introduction

The Statistics Toolbox, for use with MATLAB®, is a collection of statistical tools built on the MATLAB numeric computing environment. The toolbox supports a wide range of common statistical tasks, from random number generation, to curve fitting, to design of experiments and statistical process control. The toolbox provides two categories of tools:

- Building-block probability and statistics functions;
- Graphical, interactive tools.

The first category of tools is made up of functions that you can call from the command line or from your own applications. Many of these functions are MATLAB M-files, series of MATLAB statements that implement specialized statistics algorithms. Secondly, the toolbox provides a number of interactive tools that let you access many of the functions through a graphical user interface (GUI). Together, the GUI-based tools provide an environment for polynomial fitting and prediction, as well as probability function exploration.

The Statistics Toolbox has more than 200 M-files, divided into the following topical areas:

- **Probability Distributions.** The Statistics Toolbox supports 20 probability distributions. For each distribution there are five associated functions: probability density function (pdf), cumulative distribution function (cdf), inverse of the cumulative distribution function, random number generator, mean and variance as a function of the parameters. For most distributions, the Statistics Toolbox also provides functions for computing parameter estimates and confidence intervals.
- **Descriptive Statistics.** The Statistics Toolbox provides functions for describing the features of a data sample. These descriptive statistics include measures of location

and spread, percentile estimates and functions for dealing with data having missing values.

- **Linear Models.** In the area of linear models, the Statistics Toolbox supports one-way, two-way, and higher-way analysis of variance (ANOVA), analysis of co-variance (ANOCOVA), simple and multiple linear regression, stepwise regression, response surface prediction, ridge regression, and one-way multivariate analysis of variance (MANOVA). It supports nonparametric versions of one-and two-way ANOVA. It also supports multiple comparisons of the estimates produced by ANOVA and ANOCOVA functions.

- **Nonlinear Models.** For nonlinear models, the Statistics Toolbox provides functions for parameter estimation, interactive prediction and visualization of multidimensional nonlinear fits, and confidence intervals for parameters and predicted values. It provides functions for using classification and regression trees to approximate regression relationships.

- **Hypothesis Tests.** The Statistics Toolbox also provides functions that do the most common tests of hypothesis — t-tests, Z-tests, nonparametric tests, distribution tests, variance tests, and tests of randomness.

- **Multivariate Statistics.** The Statistics Toolbox supports methods in multivariate statistics, including principal components analysis, factor analysis, one-way multivariate analysis of variance, cluster analysis, and classical multidimensional scaling.

- **Statistical Plots.** The Statistics Toolbox adds box plots, normal probability plots, Weibull probability plots, control charts, and quantile-quantile plots to the arsenal of graphs in MATLAB. There is also extended support for polynomial curve fitting and prediction. There are functions to create scatter plots or matrices of scatter plots for grouped data, and to identify points interactively on such plots. There is a function to interactively explore a fitted regression model.

- **Statistical Process Control (SPC).** For SPC, the Statistics Toolbox provides functions for plotting common control charts and performing process capability studies.

- **Design of Experiments (DOE).** The Statistics Toolbox supports full and fractional factorial designs, response surface designs, and D-optimal designs. There are functions for generating designs, augmenting designs, and optimally assigning units with fixed covariates.

- **Hidden Markov Models.** The Statistics Toolbox provides functions for analyzing hidden Markov models — models in which you do not know all the state information. These include functions for generating random data, calculating the most probable state sequence for an observed sequence, estimating model parameters, calculating posterior state probabilities, and calculating maximum likelihood estimates for parameters.

Chapter 2

Probability Distributions

The Statistics Toolbox supports 23 probability distributions. For each distribution there are five associated functions. They are

- Probability density function or mass probability function for discrete distribution (pdf);
- Cumulative distribution function (cdf);
- Inverse of the cumulative distribution function (inv);
- Random number generator(rnd);
- Mean and variance as a function of the parameters.

For most distributions, the Statistics Toolbox also provides functions for computing parameter estimates and confidence intervals.

As a rule, the name of a function from the first five categories is formed from the distribution name and the suffix giving previously within parantheses. For example, the function `normpdf` computes the probability density function of the normal distribution.

We give in the sequel a list of of probability distribution supported by MATLAB and for each distribution its MATLAB name, that in combination with the appropriate suffix gives the function name.

Discrete probability distribution:

- Binomial – `bino`;
- Discrete uniform – `unid`;

- Geometric – `geo`;
- Hypergeometric – `hyge`;
- Negative binomial – `nbin`;
- Poisson – `poiss`.

Continuous probability distribution:

- Beta – `beta`;
- Chi-square – `chi2`;
- Noncentral chi-square – `ncx2;`
- Exponetial – `exp`;
- Extreme value – `ev;`
- Generalized extreme value distribution – `gev`;
- F – `f`;
- Noncentral F – `ncf`;
- Gamma – `gam`;
- Generalized Pareto – `gp`;
- Lognormal – `logn`;
- Normal – `norm`;
- Rayleigh – `rayl`;
- T (Student distribution) – `t`;
- Noncentral T – `nct`;
- Uniform – `uni`;
- Weibull – `wbl.`

2.1 Discrete Distributions

2.1.1 Binomial distribution

The binomial distribution models the total number of successes in repeated trials from an infinite population under the following conditions:

- Only two outcomes are possible on each of n trials.
- The probability of success for each trial is constant.
- All trials are independent of each other.

Jakob Bernoulli derived the binomial distribution in 1713. Earlier, Blaise Pascal had considered the special case where p = 1/2. The pdf (in this case the frequency or mass function) is

$$y = f(x|n,p) = \binom{n}{x} p^x q^{1-x}, \qquad x \in \{0, \dots, n\}.$$

Example 2.1.1. Plot pdf and cdf of a binomial distribution with $p = 0.2$ and $n = 10$.

Solution. The MATLAB sequence is:

```
x=0:10; y1=binopdf(x,10,0.2);
y2=binocdf(x,10,0.2);
subplot(1,2,1)
plot(x,y1,'+') title('pdf')
subplot(1,2,2)
stairs(x,y2)
title('cdf')
```

The graph is given in figure 2.1. □

Example 2.1.2. A Quality Assurance inspector tests 200 circuit boards a day. If 2% of the boards have defects, what is the probability that the inspector will find no defective boards on any given day?

```
>> binopdf(0,200,0.02)
ans =
    0.0176
```

What is the most likely number of defective boards the inspector will find?

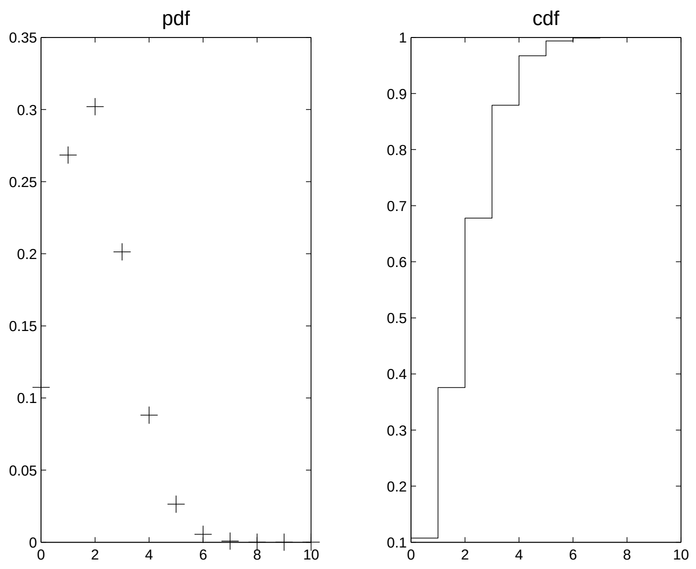


Figure 2.1: The graphs of a binomial distribution with $n = 10$ and $p = 0.2$; left - pdf, right - cdf

```
>> y = binopdf([0:200],200,0.02);
>> [x,i] = max(y);
>> i
i =
     5
```

Example 2.1.3. Suppose that a lot of 300 electrical fuses contains 5% defectives. If a sample of five fuses is testes, find the probability of observing at least one defective

Solution. It is reasonable to assume that Y, the number of defectives observed, has an approximate binomial distribution because the lot is large. Thus, the probability to observe at least one defective is

```
>> P=1-binocdf(0,5,5/100)
P =
    0.2262
```

□

Example 2.1.4. Experience has shown that 30% of all persons afflicted by a certain illness recover. A drug company has developed a new medication. Ten people with the illness were selected at random and injected with the medication; nine recovered shortly thereafter. Suppose that the medication was absolutely worthless. What is the probability that at least nine of ten injected with the medication will recover?

Solution. Let Y denote the number of people who recover and P the required probability. We have $P = P(Y >= 9) = 1 - P(Y <= 8)$, or in MATLAB:

```
>> format short g
>> 1-binocdf(8,10,0.3)
ans =
   0.00014369
```

$\square$

Example 2.1.5. If a baseball team has a 50-50 chance of winning any game, what is a reasonable range of games this team might win over a season of 162 games? We assume that a surprising result is one that occurs by chance once in a decade.

```
>> binoinv([0.05 0.95],162,0.5)
ans =
  71 91
```

This result means that in 90% of baseball seasons, a .500 team should win between 71 and 91 games. $\diamondsuit$

2.1.2 Discrete uniform distribution

The random variable X obey a discrete uniform distribution if its distribution is given by

$$X : \begin{pmatrix} k \\ N \end{pmatrix}_{k=1,\ldots,N}.$$

The discrete uniform distribution is a simple distribution that puts equal weight on the integers from one to N. Figure 2.2 gives the graph of a pdf and a cdf for a discrete uniform distribution respectively.

Example 2.1.6. What is the probability of drawing a number 20 or less from a hat with the numbers from 1 to 50 inside?

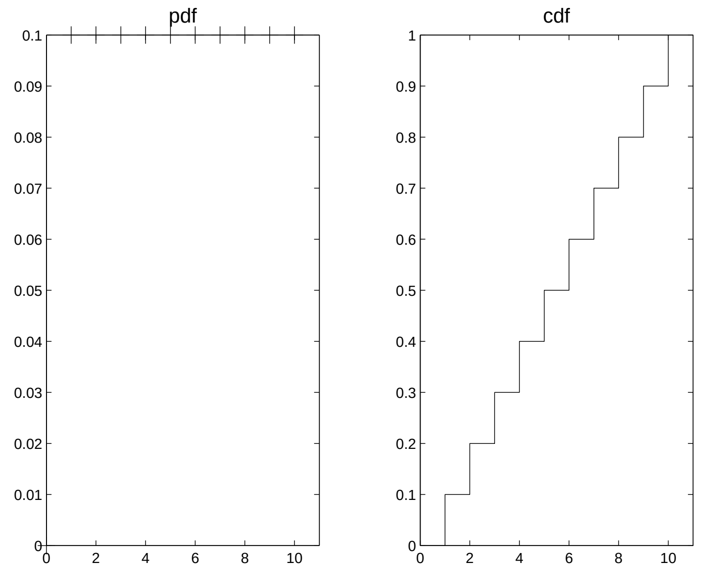


Figure 2.2: A uniform discrete pdf (left) and a cdf for $N = 10$

```
>> probability = unidcdf(20,50)
probability =
    0.4000
```

Example 2.1.7. Let us simulate throwing two dices eight times.

```
>> Z=unidrnd(6,2,8)
Z =
     4     6     2     6     3     1     5     1
     5     5     3     6     6     3     1     2
```

2.1.3 Hypergeometric distribution

The hypergeometric distribution models the total number of successes in a fixed size sample drawn without replacement from a finite population.

The distribution is discrete, existing only for nonnegative integers less than the number of samples or the number of possible successes, whichever is greater. The hypergeometric distribution differs from the binomial only in that the population is finite and the sampling from the population is without replacement.

The hypergeometric distribution has three parameters that have direct physical interpretations. M is the size of the population. K is the number of items with the desired characteristic in the population, $K \leq M$. n is the number of samples drawn. Sampling "without replacement" means that once a particular sample is chosen, it is removed from the relevant population for all subsequent selections.

The mass function for the hypergeometric distribution with parameters M, K and n is

$$y = f(x|M,K,n) = \frac{\binom{K}{x}\binom{M-K}{n-x}}{\binom{M}{n}}, \quad x = 0,1,\ldots,n, x \geq K, n-x \geq M-K.$$

The plot in Figure 2.3 shows the pdf and the cdf of an experiment taking 20 samples from a group of 1000 where there are 50 items of the desired type.

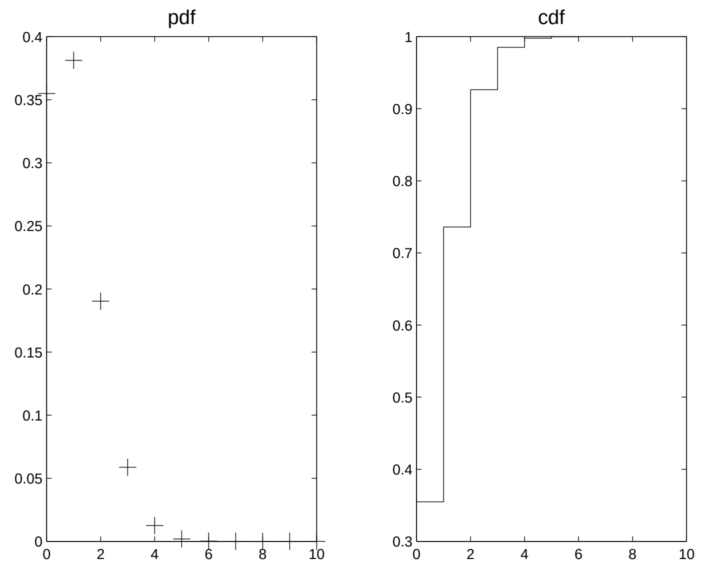


Figure 2.3: Graphs of a hypergeometric distribution with $M = 1000$, $k = 50$, $n = 20$.

Example 2.1.8. Suppose you have a lot of 100 floppy disks and you know that 20 of them are defective. What is the probability of drawing 0 through 5 defective floppy disks if you select 10 at random? What is the probability of drawing zero to two defective floppies if you select 10 at random?

```
>> p1 = hygepdf(0:5,100,20,10)
```

```
p1 =
    0.0951   0.2679    0.3182    0.2092    0.0841    0.0215
>> p2 = hygecdf(2,100,20,10)
p2 =
    0.6812
```

Example 2.1.9. Suppose you are the Quality Assurance manager for a floppy disk manufacturer. The production line turns out floppy disks in batches of 1,000. You want to sample 50 disks from each batch to see if they have defects. You want to accept 99there are no more than 10 defective disks in the batch. What is the maximum number of defective disks should you allow in your sample of 50?

```
>> x = hygeinv(0.99,1000,10,50)
x =
     3
```

What is the median number of defective floppy disks in samples of 50 disks from batches with 10 defective disks?

```
>> x = hygeinv(0.50,1000,10,50)
x =
     0
```

2.1.4 Negative binomial distribution

We say that the random variable X has a negative binomial distribution with parameters $r \in \mathbb{N}$ and $p \in (0,1)$ if its mass function is

$$y = f(x|r,p) = \binom{r+x-1}{x} p^r q^x, \qquad x \in \mathbb{N},$$

where $q = 1-p$. When r is not an integer, the binomial coefficient in the definition of the pdf is replaced by the equivalent expression

$$\frac{\Gamma(r+x)}{\Gamma(x+1)\Gamma(r)}.$$

In its simplest form, the negative binomial distribution models the number of successes before a specified number of failures is reached in an independent series of repeated identical trials. It can also be thought of as modelling the total number of trials required before a specified number of successes, thus motivating its name as the inverse of the binomial distribution. Its parameters are the probability of success in a single trial, p, and the number of failures, r. A special case of the negative binomial distribution, when $r = 1$, is the geometric distribution (see section 2.1.5), which models the number of successes before the first failure.

More generally, the r parameter can take on noninteger values. This form of the negative binomial has no interpretation in terms of repeated trials, but, like the Poisson distribution, it is useful in modelling count data. It is, however, more general than the Poisson, because the negative binomial has a variance that is greater than its mean, often making it suitable for count data that do not meet the assumptions of the Poisson distribution. In the limit, as the parameter increases to infinity, the negative binomial distribution approaches the Poisson distribution.

Example 2.1.10. A geological study indicates that an exploratory oil well drilled in a particular region should strike oil with the probability 0.2. Find the probability that the oil strike comes on the fifth well drilled and on the first five wells drilled. $\diamondsuit$

Solution. We have a negative binomial distribution with $r = 3$ and $p = 0.2$. Since $x = 2$, the required probabilities are

```
>> P1=nbinpdf(2,3,0.2)
P1 =
    0.0307
>> P2=nbincdf(2,3,0.2)
P2 =
    0.0579
```

$\square$

Figure 2.4 gives the pdf and cdf for a negative binomial distribution.

2.1.5 Geometric distribution

A random variable X is said to have a geometric probability distribution with parameter p if its mass (frequency) function is

$$y = f(x|p) = pq^{x-1}, \qquad x = 1, 2, 3, \ldots, p \in [0, 1].$$

The geometric distribution is useful for modelling the runs of consecutive successes (or failures) in repeated independent trials of a system. The geometric distribution models the number of successes before one failure in an independent succession of tests where each test results in success or failure.

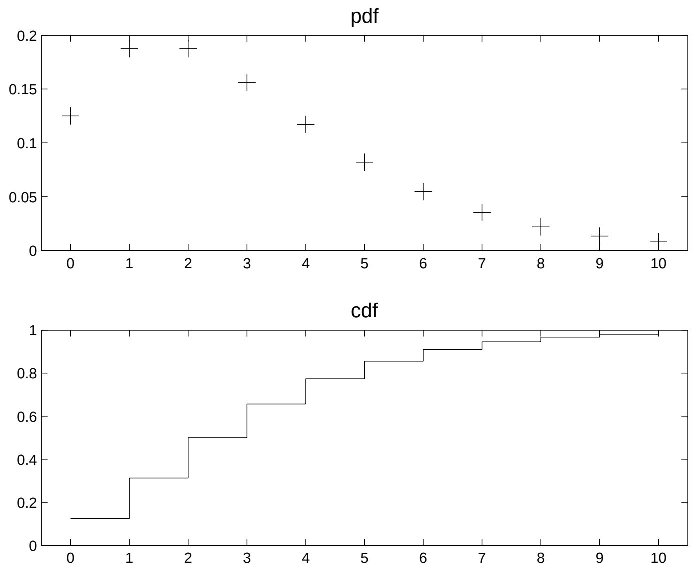


Figure 2.4: The pdf (left) and the cdf of a negative binomial distribution with $r = 3$ and $p = 0.5$.

Example 2.1.11. Suppose you toss a fair coin repeatedly. If the coin lands face up (heads), that is a success. What is the probability of observing exactly three tails before getting a heads? What is the probability of observing three or fewer tails before getting a heads? The answers for this two questions is:

```
>> p = geopdf(3,0.5)
p =
    0.0625
>> p = geocdf(3,0.5)
p =
    0.9375
```

The probability of correctly guessing the result of 10 coin tosses in a row is less than 0.001 (unless the coin is not fair).

```
>> psychic = geoinv(0.999,0.5)
psychic =
     9
```

Example 2.1.12. The example below shows the inverse method for generating random numbers from the geometric distribution.

```
>> rndgeo = geoinv(rand(2,5),0.5)
rndgeo =
     4     1     3     0     2
     0     0     2     0     0
```

We suggest to the reader to try another examples ♢

The graphs of a pdf and a cdf of a geometric distribution with $p = 0.5$ are given in figure 2.5.

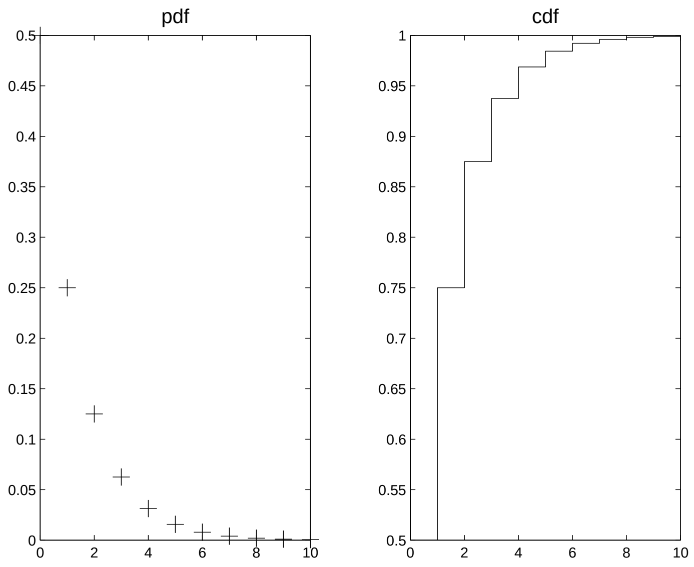


Figure 2.5: Graph of the pdf (left) and cdf for a geometric distribution with $p = 0.5$

2.1.6 Poisson distribution

A random variable X is said to have a Poisson distribution with parameter $\lambda > 0$ iff its mass function is

$$y = f(x|\lambda) = \frac{\lambda^x}{x!}e^{-\lambda}, \quad x \in \mathbb{N}.$$

The Poisson distribution is appropriate for applications that involve counting the number of times a random event occurs in a given amount of time, distance, area, etc. Sample applications that involve Poisson distributions include the number of Geiger counter clicks per second, the number of people walking into a store in an hour, and the number of flaws per 1000 feet of video tape.

The Poisson distribution is a one parameter discrete distribution that takes nonnegative integer values. The parameter, λ, is both the mean and the variance of the distribution. Thus, as the size of the numbers in a particular sample of Poisson random numbers gets larger, so does the variability of the numbers.

As Poisson showed, the Poisson distribution is the limiting case of a binomial distribution where N approaches infinity and p goes to zero while $Np = \lambda$.

The Poisson and exponential distributions are related. If the number of counts follows the Poisson distribution, then the interval between individual counts follows the exponential distribution.

Figure 2.6 gives the graphs of the pdf and the cdf of a Poisson distribution with $\lambda = 5$.

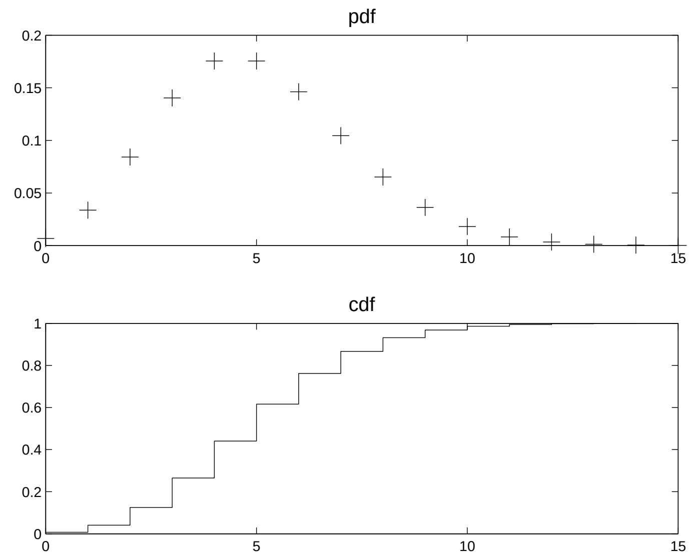


Figure 2.6: Pdf (left) and cdf for a Poisson distribution with $\lambda = 5$

Example 2.1.13. Suppose that a random system of police patrol is devised so that a patrol officer may visit a given beat location $Y = 0, 1, 2, \ldots$ times per half-hour period, with each location being visited an average of once per time period. Assume that Y possesses, approximately, a Poisson probability distribution. Calculate the probability that the patrol officer will miss a given location during a half-hour period. What is the probability that it will be visited once? Twice? At least once? $\diamondsuit$

Solution. We have $\lambda = 1$. The events that the officer visit a location 0, 1 or 2 times have the probabilities

```
p=poisspdf(0:2,1) p =
    0.3679    0.3679    0.1839
```

The probability that the location is visited at least once is $P(Y \geq 1)$, or in MATLAB

```
>> 1-poisscdf(0,1)
ans =
    0.6321
```

□

Example 2.1.14. A computer hard disk manufacturer has observed that flaws occur randomly in the manufacturing process at the average rate of two flaws in a 4 Gb hard disk and has found this rate to be acceptable. What is the probability that a disk will be manufactured with no defects?

In this example $\lambda = 2$ and $x = 0$

```
p = poisspdf(0,2) p =
    0.1353
```

Example 2.1.15. Consider a Quality Assurance department that performs random tests of individual hard disks. Their policy is to shut down the manufacturing process if an inspector finds more than four bad sectors on a disk. What is the probability of shutting down the process if the mean number of bad sectors (λ) is two?

```
>> probability = 1 - poisscdf(4,2)
probability =
    0.0527
```

About 5% of the time, a normally functioning manufacturing process will produce more than four flaws on a hard disk.

Suppose the average number of flaws (λ) increases to four. What is the probability of finding fewer than five flaws on a hard drive?

```
>> probability = poisscdf(4,4)
probability =
    0.6288
```

This means that this faulty manufacturing process continues to operate after this first inspection almost 63% of the time. ♢

Example 2.1.16. If the average number of defects (λ) is two, what is the 95th percentile of the number of defects?

```
>> poissinv(0.95,2)
ans =
     5
```

What is the median number of defects?

```
>> median_defects = poissinv(0.50,2)
median_defects =
     2
```

♢

Example 2.1.17. Generate a random sample of 10 pseudo-observations from a Poisson distribution with $\lambda = 2$.

```
>> random_sample1 = poissrnd(lambda,1,10)
random_sample1 =
    1     2     5     3     2     2     4     1     3     2
>> random_sample2 = poissrnd(lambda,[1,10])
random_sample2 =
    1     2     2     2     2     2     1     1     2     0
```

♢

2.2 Continuous Distributions

2.2.1 Beta distribution

A random variable X is said to have a beta distribution with parameters α, $\beta > 0$ if and only if its density function is

$$f(x) = \begin{cases} \dfrac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)}, & \text{for } x \in (0,1); \\ 0, & \text{elsewhere,} \end{cases}$$

where

$$B(\alpha,\beta) = \int_0^1 x^{\alpha-1}(1-x)^{\beta-1}\mathrm{d}\,x$$

is the Euler's beta function.

The graphs of beta density exhibit a large variety of shapes for various values of the two parameters α and β (see Figure 2.7). Beta distribution can be defined on an arbitrary

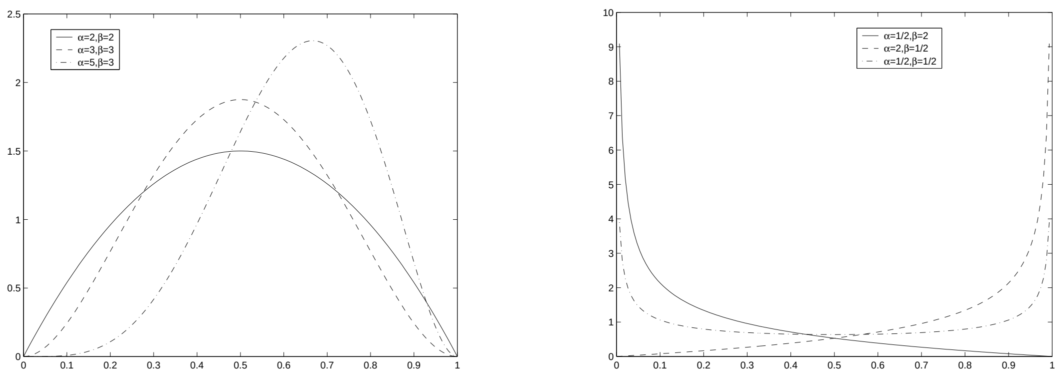


Figure 2.7: Beta density functions

interval (c, d): if $x \in (c, d)$ then $x^* = (x - c)/(d - c)$ defines a new variable such that $x^* \in (0, 1)$.

The cdf of a beta random variable is commonly called *incomplete beta function* and is denoted by

$$F(x) = \int_0^x \frac{t^{\alpha-1}(1-t)^{\beta-1}}{B(\alpha,\beta)} \mathrm{d}\, t = I_x(\alpha,\beta).$$

The value of this function is tabulated, and statistical packages implement it. For integral values of α and β $I_x(\alpha,\beta)$ is related to the binomial probability function. When $x = p$, it can be shown that

$$F(p) = \int_0^p \frac{t^{\alpha-1}(1-t)^{\beta-1}}{B(\alpha,\beta} \mathrm{d}\, t = \sum_{i=\alpha}^{n} \binom{n}{i} p^i (1-p)^{n-i},$$

where $0 < p < 1$ and $n = \alpha + \beta - 1$.

Example 2.2.1. A gasoline wholesale distributor has bulk storage tanks that hold fixed supplies and are filled every Monday. Of interest to the wholesaler is the proportion of this supply that is sold during the week. Over many weeks of observation, the distributor that this proportion could be modelled by a beta distribution with $\alpha = 4$ and $\beta = 2$. Find the probability that the wholesaler will sent at least 90% of her stock in a given week.

Let X be the proportion sold during the week. It has a $B(4, 2)$ distribution, so we compute $P(X > 0.9)$ as follows:

```
>> P=1-betacdf(0.9,4,2)
P =
    0.0815
```

It is not very likely that 90% of the stock will be sold in a given week. ♢

2.2.2 Chi-square distribution

The χ^2 distribution is a special case of the gamma distribution where $\beta = 2$ in the equation for gamma distribution (2.2.2).

The χ^2 distribution gets special attention because of its importance in normal sampling theory. If a set of n observations is normally distributed with variance σ^2, and s^2 is the sample standard deviation, then

$$\frac{(n-1)s^2}{\sigma^2} \in \chi^2(n-1)$$

The Statistics Toolbox uses the above relationship to calculate confidence intervals for the estimate of the normal parameter s^2 in the function `normfit`.

The χ^2 pdf is

$$f(x|\nu) = \begin{cases} \dfrac{x^{\frac{\nu-2}{2}} e^{\frac{-x}{2}}}{2^{\nu/2}\Gamma\left(\frac{\nu}{2}\right)}, & x > 0; \\ 0, & \text{elsewhere.} \end{cases}$$

where $\Gamma(\cdot)$ is the Gamma function, and ν is the degrees of freedom.

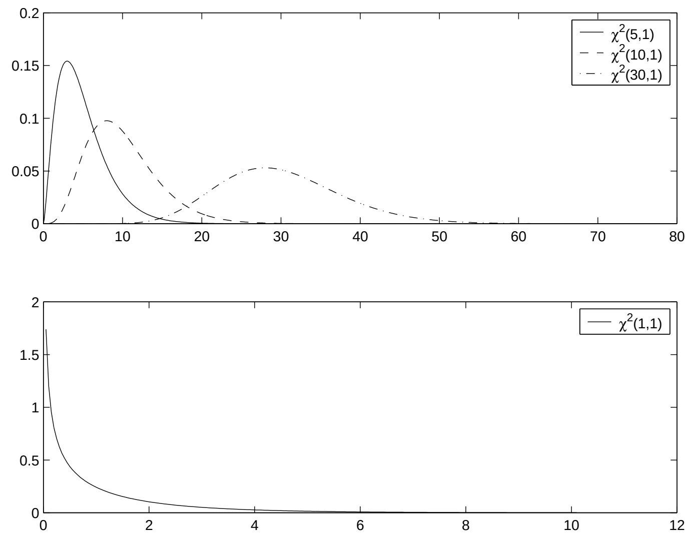


Figure 2.8: Graphs of chi-square distribution with various degrees of freedom: $\nu = 1$ (down) and $\nu = 5, 10, 30$ (up)

2.2.3 Noncentral chi-square distribution

The χ^2 distribution is actually a simple special case of the noncentral chi-square distribution. One way to generate random numbers with a χ^2 distribution (with ν degrees of freedom) is to sum the squares of ν standard normal random numbers (mean equal to zero.)

What if the normally distributed quantities have a mean other than zero? The sum of squares of these numbers yields the noncentral chi-square distribution. The noncentral chi-square distribution requires two parameters; the degrees of freedom and the noncentrality parameter. The noncentrality parameter is the sum of the squared means of the normally distributed quantities.

The noncentral chi-square has scientific application in thermodynamics and signal processing. The literature in these areas may refer to it as the Ricean or generalized Rayleigh distribution.

There are many equivalent formulas for the noncentral chi-square distribution function. One formulation uses a modified Bessel function of the first kind. Another uses the generalized Laguerre polynomials. The Statistics Toolbox computes the cumulative distribution function values using a weighted sum of χ^2 probabilities with the weights equal to the probabilities of a Poisson distribution. The Poisson parameter is one-half of the noncentrality parameter of the noncentral chi-square

$$F(x|\nu,\delta) = \sum_{j=0}^{\infty} \left(\frac{\left(\frac{1}{2}\delta\right)^j e^{\frac{\delta}{2}}}{j!} \right) P\left(\chi^2_{\nu+2j} \le j\right),$$

where δ is the noncentrality parameter.

The following commands generate a plot of the noncentral chi-square pdf (see Figure 2.9).

```
x = (0:0.1:10)';
p1 = ncx2pdf(x,4,2); p = chi2pdf(x,4);
plot(x,p,'k-',x,p1,'k-.')
legend('\chi^2(4)', 'noncentral \chi^2(4,2)')
```

2.2.4 Exponential distribution

A random variable X is said to have an *exponential distribution* with parameter $\beta > 0$ if the density function of x is

$$f(x|\beta) = \begin{cases} \dfrac{1}{\beta} e^{\frac{-x}{\beta}}, & y >= 0; \\ 0, & \text{elsewhere.} \end{cases}$$

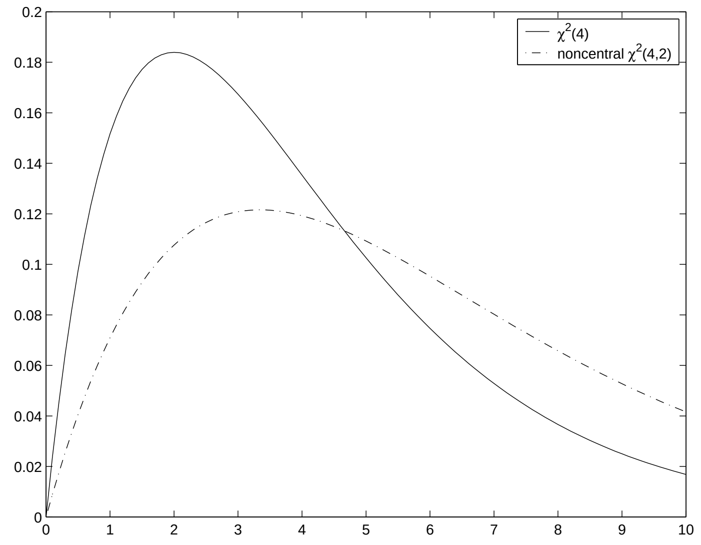


Figure 2.9: A χ^2 and a noncentral χ^2 pdf

The exponential pdf is the gamma pdf with its first parameter equal to 1.

Figure 2.10 exhibits the pdf and the cdf of an exponential distribution with $\beta = 2$.

Example 2.2.2. The median of the exponential distribution is $\beta \ln 2$. Demonstrate this fact.

```
>> bet = 10:10:60;
>> p = expcdf(log(2)*bet,bet)
p =
   0.5000   0.5000   0.5000   0.5000   0.5000   0.5000
```

What is the probability that an exponential random variable will be less than or equal to the mean, β?

```
>> bet = 1:6;
>> x = bet;
>> p = expcdf(x,bet)
p =
    0.6321   0.6321   0.6321   0.6321   0.6321   0.6321
```

The exponential distribution is often useful for modelling the length of life of technical components (electronic components, fuses and so on). It is also appropriate for modelling waiting times when the probability of waiting an additional period of time is independent

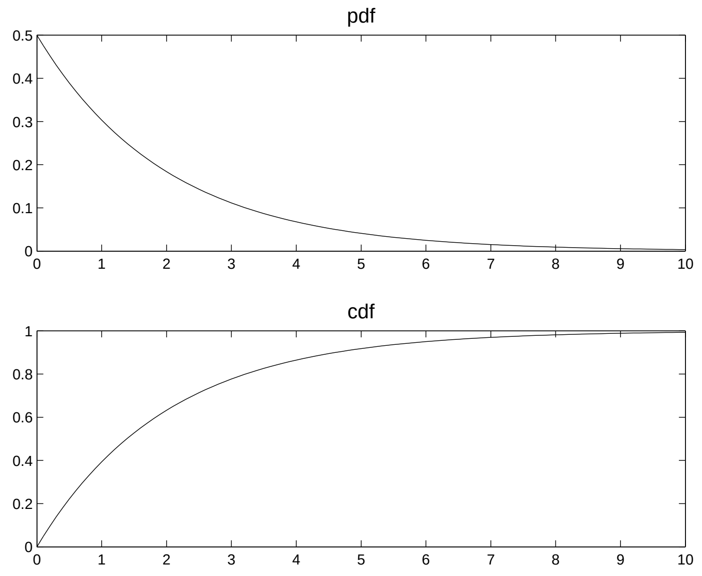


Figure 2.10: Exponential pdf (above) and cdf (below) for $\beta = 2$

of how long you've already waited. For example, the probability that a light bulb will burn out in its next minute of use is relatively independent of how many minutes it has already burned.

Example 2.2.3. Let the lifetime of light bulbs be exponentially distributed with $\beta = 700$ hours. What is the median lifetime of a bulb?

```
>> expinv(0.50,700)
ans =
  485.2030
```

So, suppose you buy a box of “700 hour” light bulbs. If 700 hours is the mean life of the bulbs, then half them will burn out in less than 500 hours. $\diamondsuit$

2.2.5 Extreme value distribution

Extreme value distributions are often used to model the smallest or largest value among a large set of independent, identically distributed random values representing measurements or observations. The extreme value distribution used in the Statistics Toolbox is appropriate for modelling the smallest value from a distribution whose tails decay exponentially fast, for example, the normal distribution. It can also model the largest value from a distribution, such as the normal or exponential distributions, by using the negative of the original values.

For example, the values generated by the following code have approximately an extreme value distribution.

```
xmin = min(randn(1000,5),[],1);
negxmax = -max(randn(1000,5),[],1);
```

Although the extreme value distribution is most often used as a model for extreme values, you can also use it as a model for other types of continuous data. For example, extreme value distributions are closely related to the Weibull distribution. If T has a Weibull distribution, then $\ln(T)$ has a type 1 extreme value distribution.

The probability density function for the extreme value distribution with location parameter μ and scale parameter σ is

$$f(x|\mu,\sigma) = \sigma^{-1}\exp\left(\frac{x-\mu}{\sigma}\right)\exp\left(-\exp\left(\frac{x-\mu}{\sigma}\right)\right).$$

If T has a Weibull distribution with parameters a and b, as described in “Weibull Distribution” on page 44, then $\ln T$ has an extreme value distribution with parameters $\mu = \ln a$ and $\sigma = 1/b$.

Example 2.2.4. The following code generates a plot of the pdf for the extreme value distribution (see Figure 2.11).

```
t = [-5:.01:2];
y = evpdf(t);
plot(t, y)
```

The extreme value distribution is skewed to the left, and its general shape remains the same for all parameter values. The location parameter, μ, shifts the distribution along the real line, and the scale parameter, σ, expands or contracts the distribution. This code plots the probability function for different combinations of μ and $sigma$.

```
x = -15:.01:5;
plot(x,evpdf(x,2,1),'-',x,evpdf(x,0,2),':',...
    x,evpdf(x,-2,4),'-.');
legend({'\mu = 2, \sigma = 1', ...
        '\mu = 0, \sigma = 2', ...
        '\mu = -2, \sigma = 4'},2)
xlabel('x')
ylabel('f(x|\mu,\sigma')
```

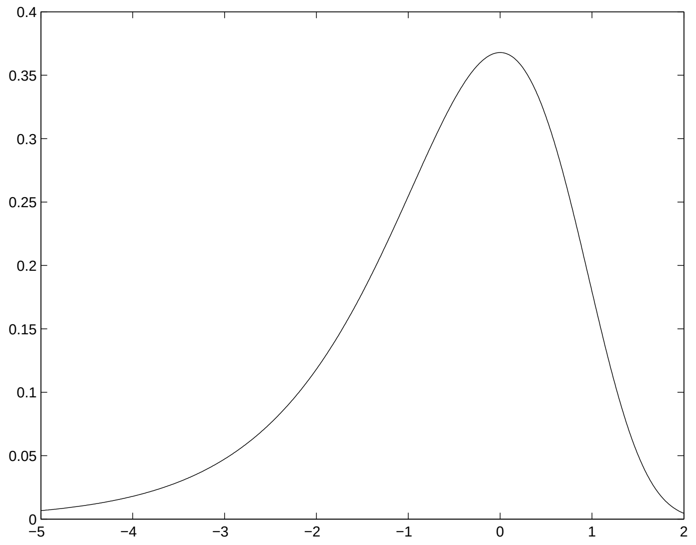


Figure 2.11: A graph of an extreme value distribution

The graph is given in Figure 2.12. ♢

2.2.6 Generalized extreme value distribution

Like the extreme value distribution, the generalized extreme value distribution is often used to model the smallest or largest value among a large set of independent, identically distributed random values representing measurements or observations. For example, you might have batches of 1000 washers from a manufacturing process. If you record the size of the largest washer in each batch, the data are known as block maxima (or minima if you record the smallest). You can use the generalized extreme value distribution as a model for those block maxima.

The generalized extreme value combines three simpler distributions into a single form, allowing a continuous range of possible shapes that includes all three of the simpler distributions. You can use any one of those distributions to model a particular dataset of block maxima. The generalized extreme value distribution allows you to "let the data decide" which distribution is appropriate.

The three cases covered by the generalized extreme value distribution are often referred to as the Types I, II, and III. Each type corresponds to the limiting distribution of block maxima from a different class of underlying distributions. Distributions whose tails

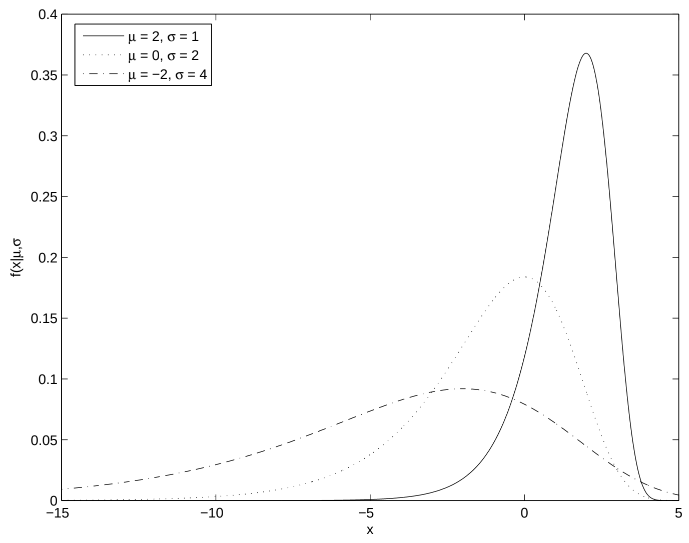


Figure 2.12: Extrem value pdf's for various combination of μ and σ

decrease exponentially, such as the normal, lead to the Type I. Distributions whose tails decrease as a polynomial, such as Students t, lead to the Type II. Distributions whose tails are finite, such as the beta, lead to the Type III.

Types I, II, and III are sometimes also referred to as the Gumbel, Frechet, and Weibull types, though this terminology can be slightly confusing. The Type I (Gumbel) and Type III (Weibull) cases actually correspond to the mirror images of the usual Gumbel and Weibull distributions, for example, as computed by the functions `evcdf` and `evfit`,or `wblcdf` and `wblfit`, respectively. Finally, the Type II (Frechet) case is equivalent to taking the reciprocal of values from a standard Weibull distribution.

The probability density function for the generalized extreme value distribution with location parameter μ, scale parameter σ, and shape parameter $k \neq 0$ is

$$f(x|k,\mu,\sigma) = \frac{1}{\sigma}\exp\left(-\left(1+k\frac{x-\mu}{\sigma}\right)^{-\frac{1}{k}}\right)\left(1+k\frac{x-\mu}{\sigma}\right)^{-1-\frac{1}{k}},$$

for

$$1+k\frac{x-\mu}{\sigma} > 0.$$

$k > 0$ corresponds to the Type II case, while $k < 0$ corresponds to the Type III case. In the limit for $k = 0$, corresponding to the Type I case, the density is

$$f(x|0,\mu,\sigma) = \sigma^{-1} \exp\left(- \exp\left(\frac{x-\mu}{\sigma} \right) - \frac{x-\mu}{\sigma} \right).$$

Example 2.2.5. The following code generates examples of probability density functions for the three basic forms of the generalized extreme value distribution.

```
x = linspace(-3,6,1000);
y1 = gevpdf(x,-.5,1,0);
y2 = gevpdf(x,0,1,0);
y3 = gevpdf(x,.5,1,0);
plot(x,y1,'-',x,y2,'--', x,y3,'-.')
legend({'K<0, Type III', 'K=0, Type I',...
        'K>0, Type II'});
```

See Figure **??**.

Notice that for $k > 0$, the distribution has zero probability density for x such that $x < -\sigma/k + \mu$. For $k < 0$, the distribution has zero probability density for $x > -\sigma/k + \mu$. In the limit for $k = 0$, there is no upper or lower bound. $\diamondsuit$

2.2.7 F distribution

The F distribution has a natural relationship with the chi-square distribution. If χ_1 and χ_2 are both chi-square with ν_1 and ν_2 degrees of freedom respectively, then the statistic F

$$F(\nu_1,\nu_2) = \frac{\frac{\chi_1}{\nu_1}}{\frac{\chi_2}{\nu_2}}$$

is F distributed.

The two parameters, ν_1 and ν_2, are the numerator and denominator degrees of freedom. That is, ν_1 and ν_2 are the number of independent pieces of information used to calculate χ_1 and χ_2, respectively.

The pdf for the F distribution is

$$f(x|\nu_1,\nu_2) = \begin{cases} \left(\frac{\nu_1}{\nu_2}\right)^{\frac{\nu_1}{2}} \dfrac{x^{\frac{\nu_1}{2}-1}}{B\left(\frac{\nu_1}{2},\frac{\nu_2}{2}\right)\left(1+\frac{\nu_1}{\nu_2}x\right)^{\frac{\nu_1+\nu_2}{2}}}, & \text{for} \quad x > 0 \\ 0, & \text{for} \quad x \le 0. \end{cases} \tag{2.2.1}$$

where $B(\cdot)$ is the Beta function.

The most common application of the F distribution is in standard tests of hypotheses in analysis of variance and regression.

The plot shows that the F distribution exists on the positive real numbers and is skewed to the right (Figure 2.13).

```
x = 0:0.01:10;
y = fpdf(x,5,3) ;
plot(x,y)
```

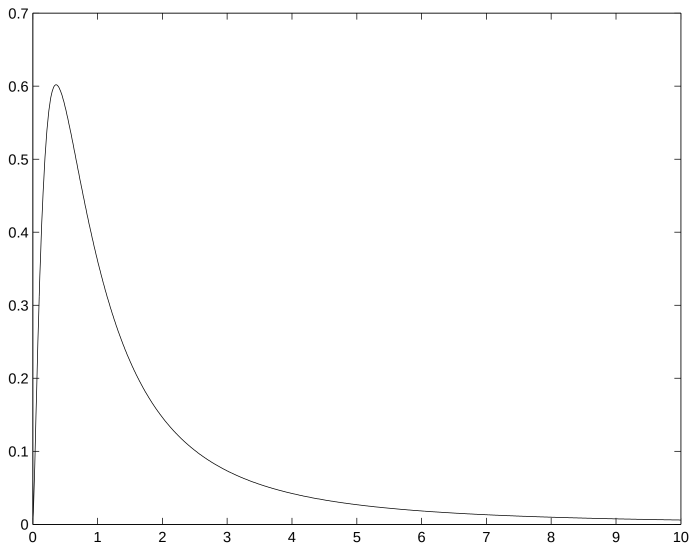


Figure 2.13: A pdf of F distribution

2.2.8 Noncentral F distribution

As with the χ^2 distribution, the F distribution is a special case of the noncentral F distribution. The F distribution is the result of taking the ratio of χ_2 random variables each divided by its degrees of freedom.

If the numerator of the ratio is a noncentral chi-square random variable divided by its degrees of freedom, the resulting distribution is the noncentral F distribution.

The main application of the noncentral F distribution is to calculate the power of a hypothesis test relative to a particular alternative.

Similar to the noncentral χ^2 distribution, the toolbox calculates noncentral F distribution probabilities as a weighted sum of incomplete beta functions using Poisson probabilities as the weights.

$$\sum_{j=0}^{\infty} \left(\frac{\left(\frac{1}{2}\delta\right)^j}{j!} e^{\frac{\delta}{2}} \right) I \left(\frac{\nu_1 x}{\nu_2 + \nu_1 x} | \frac{\nu_1}{2} + j, \frac{\nu_2}{2} \right) .$$

$I(x|\alpha, \beta)$ is the incomplete beta function with parameters α and β, and δ is the noncentrality parameter.

The following commands generate a plot of the noncentral F pdf (Figure 2.14).

```
x = (0.01:0.1:10.01)';
p1 = ncfpdf(x,5,20,10);
p = fpdf(x,5,20);
plot(x,p,'k-',x,p1,'k-.')
legend('F(5,20)', 'noncentral F(5,20,10)')
```

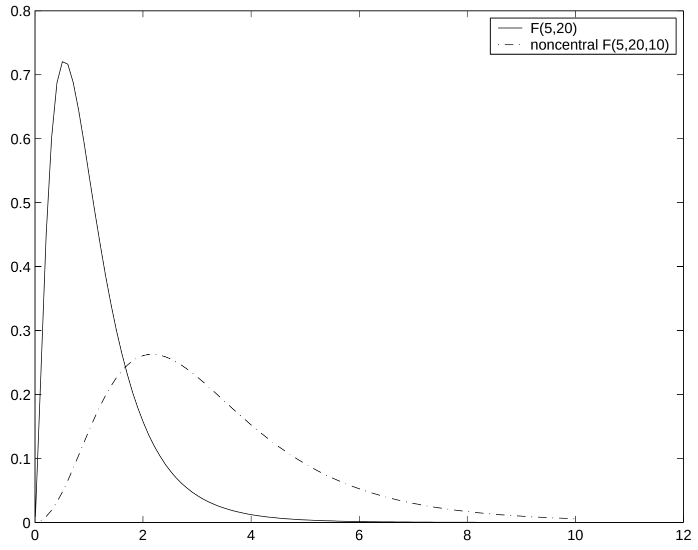


Figure 2.14: Pdf for an F and a noncentral F

2.2.9 Gamma distribution

A random variable X is said to have a *gamma distribution* with parameters α and β iff the pdf of X is

$$f(x|\alpha,\beta) = \begin{cases} \dfrac{x^{\alpha-1}e^{-\frac{x}{\beta}}}{\beta^{\alpha}\Gamma(\alpha)}, & x \geq 0; \\ 0, & \text{elsewhere.} \end{cases} \tag{2.2.2}$$

where

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

is the gamma function.

Gamma density functions for $\alpha = 1, 2$ and 4 and $\beta = 1$ are given in Figure 2.15. The parameter α is called the *shape parameter* since it determines the shape of the pdf curve. β is called the *scale parameter*. See Figure 2.16 shows the influence of β for fixed α.

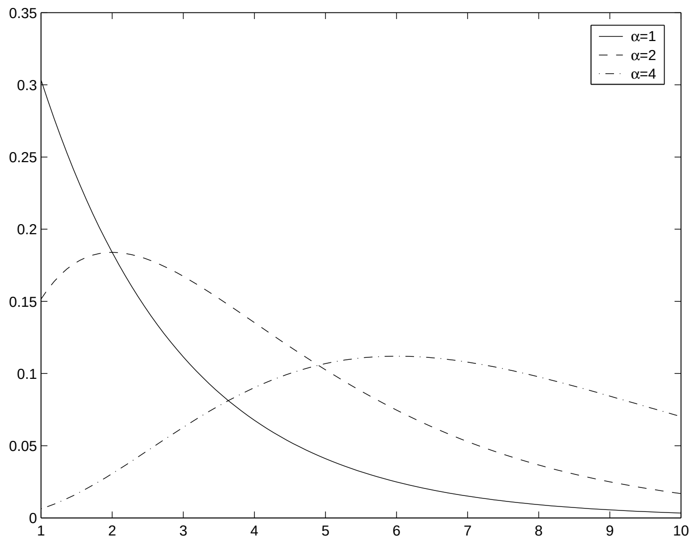


Figure 2.15: Gamma density functions for $\beta = 1$

Example 2.2.6. Four-week summer rainfall totals in a section of the midwest united states have approximately a gamma distribution with $\alpha = 1.6$ and $\beta = 2.0$. Find the probability to have an amount of rainfall between 3 and 57. What is the median of the rainfall?

```
>> gamcdf(7,1.6,2)-gamcdf(3,1.6,2)
ans =
```

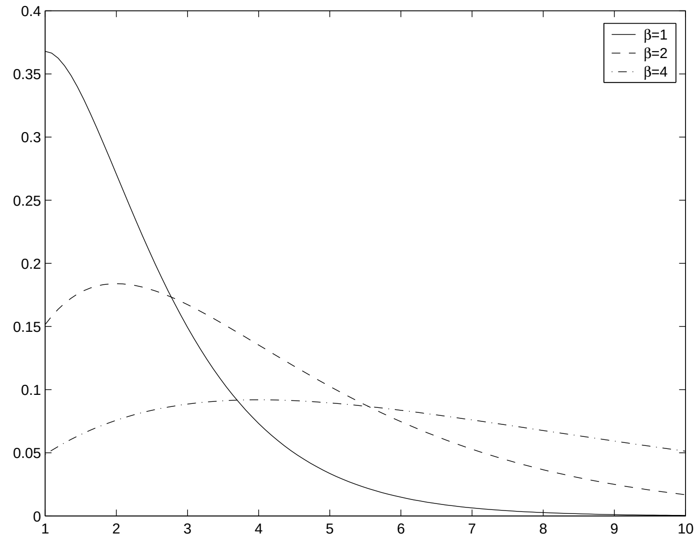


Figure 2.16: Gamma density functions, $\alpha = 2$

```
    0.3430
>> gaminv(0.5,1.6,2)
ans =
    2.5636
```

Thus an amount less than 2.5636 has a probability of 1/2. ♢

Example 2.2.7. Annual incomes for heads of household in a section of a city have approximately a gamma distribution with $\alpha = 1000$ and $\beta = 20$. Find the mean and the variance of these incomes. Would you expect to find many incomes in excess of $40,000 in this section of the city?

```
>> alpha=1000; beta=20;
>> mean=alpha*beta
mean =
       20000
>> variance=alpha*beta^2
variance =
      400000
>> P=1-gamcdf(40000,alpha,beta)
P =
     0
```

So, the probability is negligible. $\diamondsuit$

Two special cases of of gamma-distributed random variables merit particular consideration. For $\alpha = \nu/2$ and $\beta = 2$ we have a *chi-square distribution* with ν degrees of freedom. For $\alpha = 1$ we obtain an *exponential distribution*.

2.2.10 Generalized Pareto distribution

Like the exponential distribution, the generalized Pareto distribution is often used to model the tails of another distribution. For example, you might have washers from a manufacturing process. If random influences in the process lead to differences in the sizes of the washers, a standard probability distribution, such as the normal, could be used to model those sizes. However, while the normal distribution might be a good model near its mode, it might not be a good fit to real data in the tails and a more complex model might be needed to describe the full range of the data. On the other hand, only recording the sizes of washers larger (or smaller) than a certain threshold means you can fit a separate model to those tail data, which are known as exceedences. You can use the generalized Pareto distribution in this way, to provide a good fit to extremes of complicated data.

The generalized Pareto distribution allows a continuous range of possible shapes that includes both the exponential and Pareto distributions as special cases. You can use either of those distributions to model a particular dataset of exceedences. The generalized extreme value distribution allows you to “let the data decide” which distribution is appropriate.

The generalized Pareto distribution has three basic forms, each corresponding to a limiting distribution of exceedence data from a different class of underlying distributions. Distributions whose tails decrease exponentially, such as the normal, lead to a generalized Pareto shape parameter of zero. Distributions whose tails decrease as a polynomial, such as Students t, lead to a positive shape parameter. Distributions whose tails are finite, such as the beta, lead to a negative shape parameter.

The probability density function for the generalized Pareto distribution with shape parameter $k \neq 0$, scale parameter σ, and threshold parameter θ,is

$$f(x|k,\sigma,\theta) = \frac{1}{\sigma}\left(1 + k\frac{x-\theta}{\sigma}\right)^{-1-\frac{1}{k}},$$

for $\theta < x$, when $k > 0$, or for $\theta < x < -\sigma/k$ when $k < 0$. In the limit for $k = 0$, the density is

$$f(x|0,\sigma,\theta) = \frac{1}{\sigma}\exp\left(-\frac{x-\theta}{\sigma}\right),$$

for $\theta < x$. If $k = 0$ and $\theta = 0$, the generalized Pareto distribution is equivalent to the exponential distribution. If $k > 0$ and $\theta = \sigma$, the generalized Pareto distribution is equivalent to the Pareto distribution.

Example 2.2.8. The following code generates examples of the probability density functions for the three basic forms of the generalized Pareto distribution.

```
x = linspace(0,10,1000);
y1 = gppdf(x,-.25,1,0);
y2 = gppdf(x,0,1,0);
y3 = gppdf(x,1,1,0);
plot(x,y1,'-', x,y2,'--', x,y3,'-.')
legend({'K<0' 'K=0' 'K>0'});
```

Notice that for $k < 0$, the distribution has zero probability density for $x > -\sigma/k$, while for $k \geq 0$, there is no upper bound. $\diamondsuit$

2.2.11 Lognormal distribution

The normal and lognormal distributions are closely related. If X is distributed lognormal with parameters μ and σ^2, then $\ln X$ is distributed normal with parameters μ and σ^2.

The lognormal distribution is applicable when the quantity of interest must be positive, since $\ln X$ exists only when the random variable X is positive. Economists often model the distribution of income using a lognormal distribution.

The lognormal pdf is

$$f(x|\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{\ln x - \mu}{2\sigma^2}}, \qquad x > 0.$$

Example 2.2.9. Suppose the income of a family of four in the United States follows a lognormal distribution with $\mu = \ln(20,000)$ and $\sigma^2 = 1$. Plot the income density. What is the probability that the income be larger than 60000$.

The plot sequence is

```
x = (10:1000:125010)';
y = lognpdf(x,log(20000),1.0);
plot(x,y)
set(gca,'xtick',[0 30000 60000 90000 120000])
set(gca,'xticklabel',str2mat('0','$30,000','$60,000',...
'$90,000','$120,000'))
```

The graph is given in Figure 2.17. The required probability is computed with

```
P=1-logncdf(60000,log(20000),1.0)
P =
    0.1360
```

See the tail in Figure 2.17. ♢

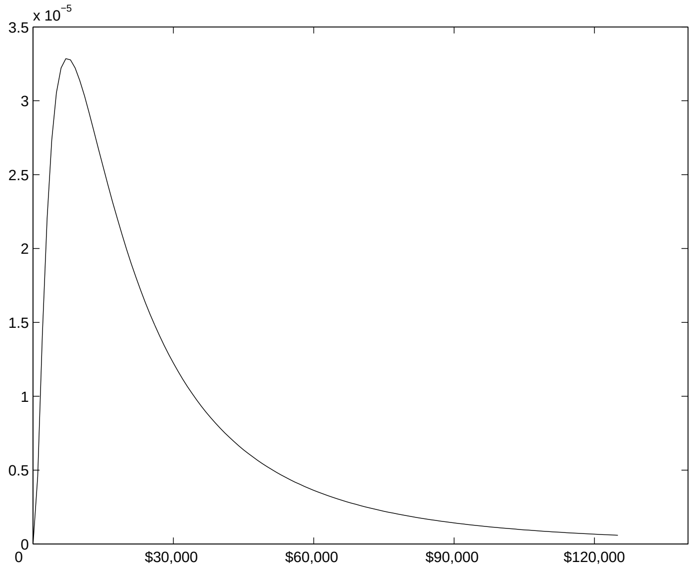


Figure 2.17: A lognormal pdf (Example 2.2.9)

2.2.12 Normal distribution

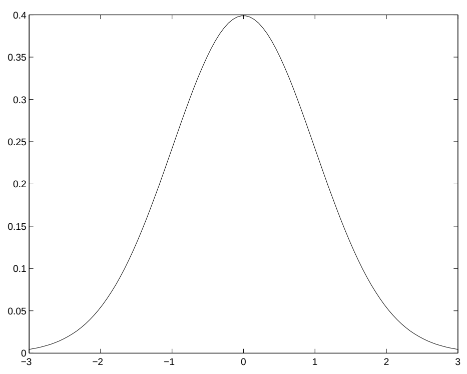
A random variable X is said to have a normal probability distribution with parameters $\mu \in \mathbb{R}$, $\sigma > 0$ iff the pdf of X is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}, \quad x \in \mathbb{R}.$$

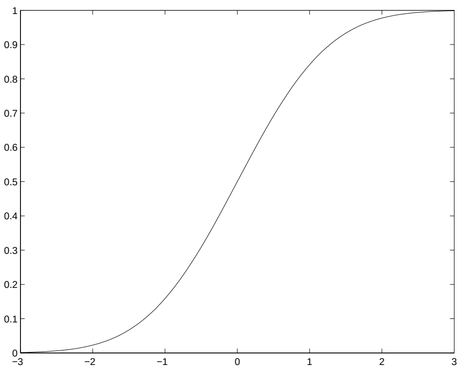
See Figure 2.18 for a graph of a standard normal pdf and a cdf.

Example 2.2.10. Check the 3σ rule for a given normal distribution. ♢

Solution. The MATLAB code is



(a) pdf



(b) cdf

Figure 2.18: Pdf and cdf for a standard normal distribution

```
function p=check_3_sigma(m,sigma)
v=normcdf([m-3*sigma,m+3*sigma],m,sigma);
p=v(2)-v(1);
```

We try it for $\mu = 0$, $\sigma = 1$

```
>> check_3_sigma(0,1)
ans =
    0.9973
```

$\square$

Example 2.2.11. Find an interval that contains 95% of the values from a standard normal distribution.

```
>> x = norminv([0.025 0.975],0,1)
x =
   -1.9600    1.9600
```

Note the interval `x` is not the only such interval, but it is the shortest.

```
>> xl = norminv([0.01 0.96],0,1)
xl =
   -2.3263    1.7507
```

The interval `xl` also contains 95% of the probability, but it is longer than `x`. $\diamondsuit$

2.2.13 Rayleigh distribution

The Rayleigh distribution is a special case of the Weibull distribution. If A and B are the parameters of the Weibull distribution, then the Rayleigh distribution with parameter b is equivalent to the Weibull distribution with parameters $A = \sqrt{2}b$ and $B = 2$.

If the component velocities of a particle in the x and y directions are two independent normal random variables with zero means and equal variances, then the distance the particle travels per unit time is distributed Rayleigh.

The Rayleigh pdf is

$$f(x|b) = \frac{x}{b^2} e^{-\frac{x^2}{2b^2}}.$$

The following commands generate a plot of the Rayleigh pdf (Figure 2.19).

```
x = [0:0.01:2];
p = raylpdf(x,0.5);
plot(x,p)
```

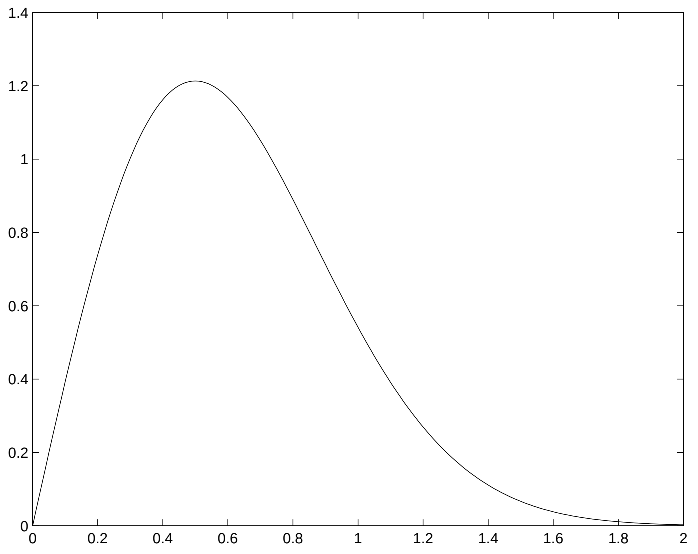


Figure 2.19: Plot of the Rayleigh pdf

2.2.14 T (Student) distribution

The t distribution is a family of curves depending on a single parameter ν (the degrees of freedom). As ν goes to infinity, the t distribution converges to the standard normal distribution.

W. S. Gossett discovered the distribution through his work at the Guinness brewery. At that time, Guinness did not allow its staff to publish, so Gossett used the pseudonym Student. If x and σ are the mean and standard deviation of an independent random sample of size n from a normal distribution with mean μ and $\sigma^2 = n$, then

$$T(\nu) = \frac{x - \mu}{s}, \quad \nu = n - 1.$$

Students t pdf is

$$f(x|\nu) = \frac{1}{\sqrt{2\pi}} \frac{\Gamma\left(\frac{\nu+1}{1}\right)}{\Gamma\left(\frac{\nu}{2}\right)} \left(1 + \frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}},$$

where $\Gamma(\cdot)$ is the Gamma function.

The plot compares the t distribution with $\nu = 5$ (solid line) to the shorter tailed, standard normal distribution (dashed line) (Figure 2.20).

```
x = -5:0.1:5;
y = tpdf(x,5);
z = normpdf(x,0,1);
plot(x,y,'-',x,z,'-.')
legend('T(5)', 'N(0,1)')
```

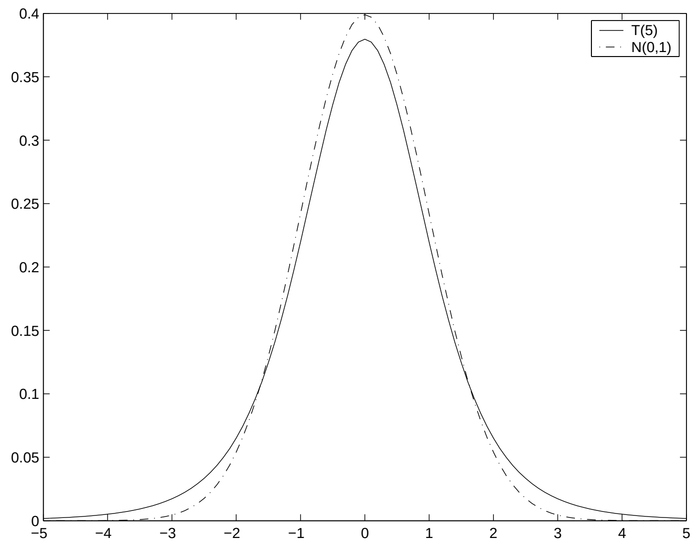


Figure 2.20: T and norma pdf

2.2.15 Noncentral T distribution

The noncentral T distribution is a generalization of the familiar Students T distribution. Recall that the ordinary Students T distribution is defined as follows. If $\bar{X}$ and S are the sample mean and standard deviation of an independent random sample of size n from a normal distribution with mean μ_0 and standard deviation σ, then

$$T(\nu) = \frac{\bar{(}X) - \mu_0}{S/\sqrt{n}},$$

where $\nu = n - 1$, has the ordinary Students t distribution with ν degrees of freedom.

Now, suppose that the true mean of the distribution of X is μ, rather than the hypothesized value μ_0. Then the ratio on the right-hand side of the preceding equation has a noncentral T distribution with a noncentrality parameter δ equal to

$$\delta = \frac{\mu - \mu_0}{\sigma/\sqrt{n}}.$$

δ is the normalized difference between the true mean and the hypothesized mean.

The noncentral T distribution enables you to determine the probability of detecting a difference between μ and μ_0 in a t test. This probability is the power of the test. The power increases as the difference $\mu - \mu_0$ increases, and also as the sample size increases.

The most general representation of the noncentral T distribution is quite complicated. Johnson and Kotz give a formula for the probability that a noncentral T variate falls in the range $[-t, t]$.

$$P(-t < X < t)(\nu, \delta) = \sum_{j=0}^{\infty} \left(\frac{\left(\frac{\delta^2}{2}\right)^j}{j!} e^{\frac{\delta^2}{2}} \right) I \left(\frac{x^2}{\nu + x^2} \middle| \frac{1}{2} + j, \frac{\nu}{2} \right),$$

where $I(x|a, b)$ is the incomplete beta function with parameters a and b, δ is the noncentrality parameter, and ν is the degrees of freedom.

Example 2.2.12. The following commands generate a plot of the noncentral t pdf.

```
x = (-5:0.1:5)';
p1 = nctcdf(x,10,1);
p = tcdf(x,10);
plot(x,p,'--',x,p1,'-')
legend('T','noncentral T',0)
```

The graph is given in figure 2.21. $\diamondsuit$

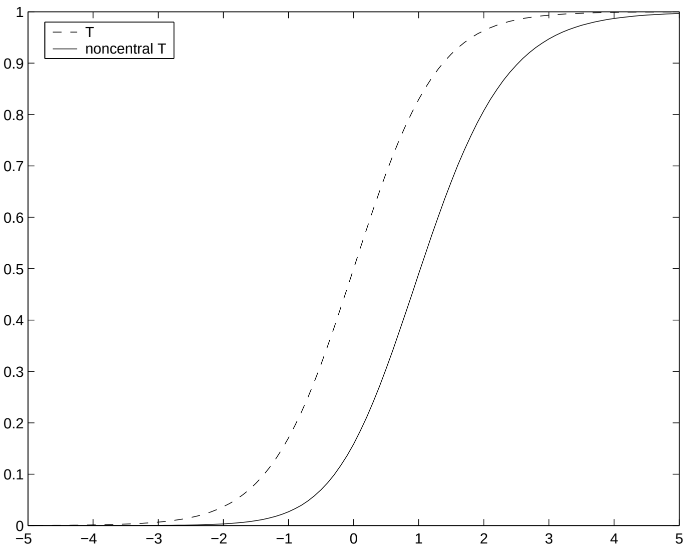


Figure 2.21: Noncentral T distribution

2.2.16 Uniform distribution

A random variable X is said to have a continuous *uniform probability distribution* on the interval (a, b) if and only if the probability density function (pdf) of X is

$$f(x|a,b) = \begin{cases} \dfrac{1}{b-a}, & \text{if } a \geq x \geq b; \\ 0, & \text{otherwise.} \end{cases}$$

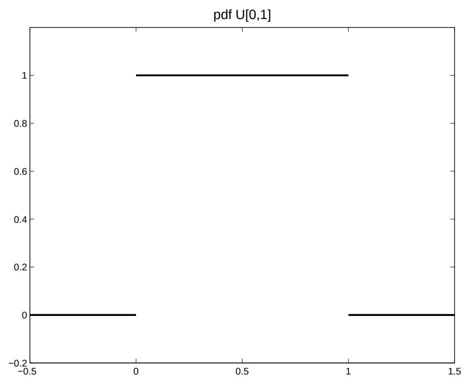
The uniform cdf of X is

$$F(x|a,b) = \begin{cases} \dfrac{x-a}{b-a}, & \text{if } a \geq x \geq b; \\ 0, & \text{otherwise.} \end{cases}$$

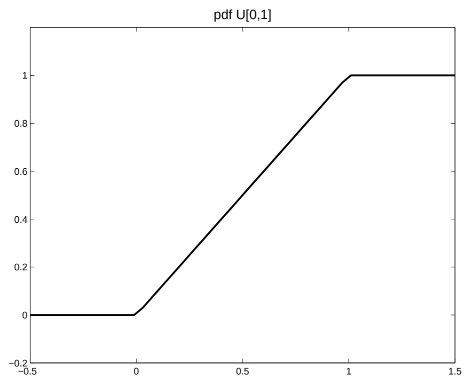
Figure 2.22 gives the graphs of pdf and cdf of a $U[0, 1]$ distribution.

Example 2.2.13. What is the probability that an observation from a uniform distribution with $a = -1$ and $b = 1$ will be less than 0.75?

```
>> probability = unifcdf(0.75,-1,1)
probability =
    0.8750
```



(a) pdf



(b) cdf

Figure 2.22: The pdf and the cdf for a $U[0, 1]$ distribution

Example 2.2.14. What is the 99th percentile of the uniform distribution between -1 and 1?

```
>> percentile = unifinv(0.99,-1,1)
percentile =
    0.9800
```

2.2.17 Weibull distribution

Waloddi Weibull offered the distribution that bears his name as an appropriate analytical tool for modeling the breaking strength of materials. Current usage also includes reliability and lifetime modeling. The Weibull distribution is more flexible than the exponential for these purposes.

To see why, consider the hazard rate function (instantaneous failure rate). If $f(t)$ and $F(t)$ are the pdf and cdf of a distribution, then the hazard rate is

$$h(t) = \frac{f(t)}{1 - F(t)}.$$

Substituting the pdf and cdf of the exponential distribution for f(t) and F(t) above yields a constant. The example below shows that the hazard rate for the Weibull distribution can vary.

The Weibull pdf is

$$f(x|a,b) = \begin{cases} ba^{-b}x^{b-1}e^{\left(\frac{x}{a}\right)^b}, & \text{for } x > 0; \\ 0, & \text{otherwise.} \end{cases}$$

Example 2.2.15. The exponential distribution has a constant hazard function, which is not generally the case for the Weibull distribution. The plot (see Figure 2.23) shows the hazard functions for exponential (dashed line) and Weibull (solid line) distributions having the same mean life. The Weibull hazard rate here increases with age (a reasonable assumption).

```
t = 0:0.1:4.5;
h1 = exppdf(t,0.6267) ./ (1-expcdf(t,0.6267));
h2 = weibpdf(t,2,2) ./ (1-weibcdf(t,2,2));
plot(t,h1,'--',t,h2,'-')
```

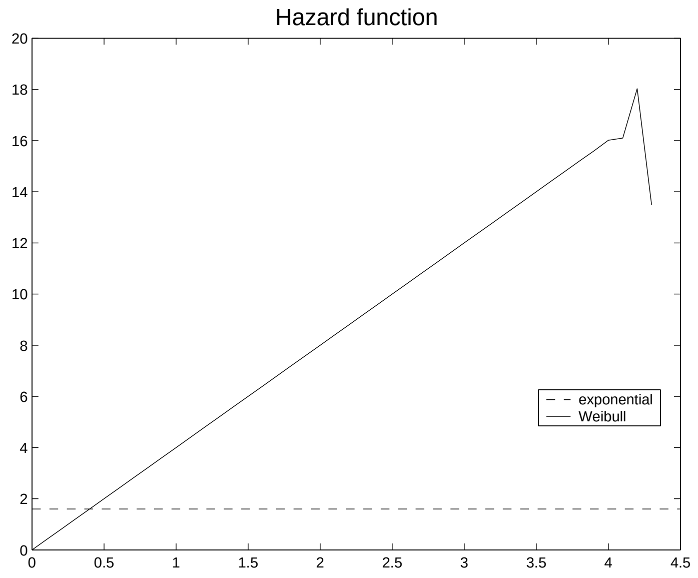


Figure 2.23: Hazard function for exponential

Chapter 3

Descriptive Statistics

3.1 Measures of Central Tendency (Location)

The purpose of measures of central tendency is to locate the data values on the number line. Another term for these statistics is measures of location.

The table gives the function names and descriptions.

Function Name	Description
`geomean`	Geometric mean
`harmmean`	Harmonic mean
`mean`	Arithmetic average (in MATLAB)
`median`	50th percentile (in MATLAB)
`mode`	Most frequent value (in MATLAB)
`trimmean`	Trimmed mean

The average is a simple and popular estimate of location. If the data sample comes from a normal distribution, then the sample average is also optimal (MVUE of μ).

Unfortunately, outliers, data entry errors, or glitches exist in almost all real data. The sample average is sensitive to these problems. One bad data value can move the average away from the center of the rest of the data by an arbitrarily large distance.

The median and trimmed mean are two measures that are resistant (robust) to outliers. The median is the 50th percentile of the sample, which will only change slightly if you add a large perturbation to any value. The idea behind the trimmed mean is to ignore a small percentage of the highest and lowest values of a sample when determining the center of the sample.

The geometric mean and harmonic mean, like the average, are not robust to outliers. They are useful when the sample is distributed lognormal or heavily skewed.

The following example shows the behavior of the measures of location for a sample with one outlier.

```
>> x = [ones(1,6) 100]
x =
     1     1     1     1     1     1   100
>> locate = [geomean(x) harmmean(x) mean(x) median(x)...
        trimmean(x,25)]
locate =
    1.9307    1.1647   15.1429    1.0000    1.0000
```

You can see that the mean is far from any data value because of the influence of the outlier. The median and trimmed mean ignore the outlying value and describe the location of the rest of the data values.

3.2 Measures of Dispersion

The purpose of measures of dispersion is to find out how spread out the data values are on the number line. Another term for these statistics is measures of spread.

The table gives the function names and descriptions.

Function	Description
Name	
`iqr`	Interquartile Range
`mad`	Mean Absolute Deviation
`range`	Range
`std`	Standard deviation (in MATLAB)
`var`	Variance (in MATLAB)

The range (the difference between the maximum and minimum values) is the simplest measure of spread. But if there is an outlier in the data, it will be the minimum or maximum value. Thus, the range is not robust to outliers.

The standard deviation and the variance are popular measures of spread that are optimal for normally distributed samples. The sample variance is the MVUE of the normal parameter s2. The standard deviation is the square root of the variance and has the desirable property of being in the same units as the data. That is, if the data is in meters, the standard deviation is in meters as well. The variance is in meters2, which is more difficult to interpret.

Neither the standard deviation nor the variance is robust to outliers. A data value that is separate from the body of the data can increase the value of the statistics by an arbitrarily large amount.

The Mean Absolute Deviation (MAD) is also sensitive to outliers. But the MAD does not move quite as much as the standard deviation or variance in response to bad data.

The Interquartile Range (IQR) is the difference between the 75th and 25th percentile of the data. Since only the middle 50% of the data affects this measure, it is robust to outliers.

The following example shows the behavior of the measures of dispersion for a sample with one outlier.

```
>> x = [ones(1,6) 100]
x =
     1     1     1     1     1     1   100
>> stats = [iqr(x) mad(x) range(x) std(x)]
stats =
         0   24.2449   99.0000   37.4185
```

3.3 Functions for Data with Missing Values (NaNs)

Most real-world data sets have one or more missing elements. It is convenient to code missing entries in a matrix as NaN (Not a Number).

Here is a simple example.

```
>> m = magic(3) ; m([ 1 5] ) = [NaN NaN]
m =
   NaN     1     6
     3   NaN     7
     4     9     2
```

Any arithmetic operation that involves the missing values in this matrix yields NaN, as below.

```
>> sum(m)
ans =
   NaN   NaN    15
```

Removing cells with NaN would destroy the matrix structure. Removing whole rows that contain NaN would discard real data. Instead, the Statistics Toolbox has a variety of functions that are similar to other MATLAB functions, but that treat NaN values as missing and therefore ignore them in the calculations (see Table 3.1.

`nanmax`	Maximum ignoring NaNs
`nanmean`	Mean ignoring NaNs
`nanmedian`	Median ignoring NaNs
`nanmin`	Minimum ignoring NaNs
`nanstd`	Standard deviation ignoring NaNs
`nansum`	Sum ignoring NaNs

Table 3.1: NaN Functions

```
>> nansum(m)
ans =
     7    10    15
```

In addition, other Statistics Toolbox functions operate only on the numeric values, ignoring NaNs. These include `iqr`, `kurtosis`, `mad`, `prctile`, `range`, `skewness`, and `trimmean`.

3.4 Function for Grouped Data

As you saw in the previous section, the descriptive statistics functions can compute statistics on each column in a matrix. Sometimes, however, you may have your data arranged differently so that measurements appear in one column or variable, and a grouping code appears in a second column or variable. Although the MATLAB syntax makes it simple to apply functions to a subset of an array, in this case it is simpler to use the `grpstats` function.

The `grpstats` function can compute the mean, standard error of the mean, and count (number of observations) for each group defined by one or more grouping variables. If you supply a significance level, it also creates a graph of the group means with confidence intervals.

The following example generates a 100×3 matrix of normal random numbers with mean 1, 2, 3 for each column respectively. As grouping vector we use a uniform discrete random vector with $N = 4$.

```
g=unidrnd(4,100,1);
t=[1:3]; t=t(ones(100,1),:);
x=normrnd(t,1);
[am,sd,fr,name]=grpstats(x,g);
[name num2cell([am, sd, fr])]
```

The results are

```
ans =
  Columns 1 through 5
    '1'    [1.3703]    [2.2371]    [3.0534]    [0.2066]
    '2'    [0.8865]    [2.4651]    [2.7960]    [0.2825]
    '3'    [1.0790]    [2.0968]    [3.0633]    [0.1750]
    '4'    [1.2124]    [2.1185]    [2.7007]    [0.1969]
  Columns 6 through 10
    [0.2004]    [0.1761]    [27]    [27]    [27]
    [0.2760]    [0.2160]    [16]    [16]    [16]
    [0.1800]    [0.1910]    [30]    [30]    [30]
    [0.1774]    [0.2406]    [27]    [27]    [27]
```

A call `grpstats(x,group,alpha)` plot a graph of means and confidence interval for each group.

3.5 Percentiles

An alternative to measure of location and measure of spread is to compute a reasonable number of the sample percentiles. This provides information about the shape of the data as well as its location and spread.

The example shows the result of looking at every quartile of a sample containing a mixture of two distributions.

```
x = [normrnd(4,1,1,100) normrnd(6,0.5,1,200)];
p = 100*(0:0.25:1);
y = prctile(x,p);
z = [p;y]

z =
         0   25.0000   50.0000   75.0000  100.0000
    1.1572    4.5843    5.7357    6.2353    7.3233
```

3.6 Data tabulation

The function `tabulate` computes a frequency table.

The form `table = tabulate(x)` takes a vector of positive integers, `x`, and returns a matrix, `table`.

The first column of table contains the values of `x`. The second contains the number of instances of this value. The last column contains the percentage of each value.

`TABLE = tabulate(ARRAY)`, where `ARRAY` is a character array or a cell array of strings, returns `TABLE` as a cell array. The first column contains the unique string values in `ARRAY`. The other two columns contain the same information as for a vector input.

`tabulate` with no output arguments displays a formatted table in the command window.

```
>> a=unidrnd(6,1,10)
a =
    1    5    3    6    3    3    6    4    2    5
>> tabulate(a)
  Value    Count    Percent
      1        1     10.00%
      2        1     10.00%
      3        3     30.00%
      4        1     10.00%
      5        2     20.00%
      6        2     20.00%
```

The function `crosstab` perform cross tabulation of several vectors. The syntax of this function is

```
table = crosstab(col1,col2)
table = crosstab(col1,col2,col3,...)
[table,chi2,p] = crosstab(col1,col2)
[table,chi2,p,label] = crosstab(col1,col2)
```

`table = crosstab(col1,col2,col3,...)` accepts vectors containing integer or noninteger values, character arrays, or cell arrays of strings. `crosstab` implicitly assigns a positive integer group number to each distinct value in `col1`, `col2` ..., and creates a cross-tabulation using those numbers. It returns `table` as an `n`-dimensional array, where `n` is the number of arguments you supply. The value of `table(i,j,k,...)` is the count of all instances where `col1 = i`, `col2 = j`, `col3 = k`, and so on.

For two input parameters one obtains a contingency table.

`[table,chi2,p] = crosstab(col1,col2)` also returns the chi-square statistic, `chi2`, for testing the independence of the rows and columns of table. The scalar `p` is the significance level of the test. Values of `p` near zero cast doubt on the assumption of independence of the rows and columns of table.

`[table,chi2,p,label] = crosstab(col1,col2)` also returns a cell array label that has one column for each input argument. The value in `label(i,j)` is the value of `colj` that defines group `i` in the `j`th dimension.

Example 3.6.1. We generate 2 columns of 50 discrete uniform random numbers. The first column has numbers from 1 to 3. The second has only the numbers 1 and 2. The two columns are independent so we would be surprised if p were near zero.

```
r1 = unidrnd(3,50,1);
r2 = unidrnd(2,50,1);
[table,chi2,p] = crosstab(r1,r2)
table =
     6     5
     8    10
    12     9
chi2 =
    0.6628
p =
    0.7179
```

The result, 0.1242, is not a surprise. A very small value of `p` would make us suspect the ”randomness” of the random number generator. ♢

3.7 Graphical data presentation

3.7.1 Histograms

`hist` displays a histogram plot. A histogram shows the distribution of data values.

`n = hist(Y)` bins the elements in `Y` into 10 equally spaced containers and returns the number of elements in each container. If `Y` is a matrix, `hist` works down the columns.

`n = hist(Y,x)` where `x` is a vector, returns the distribution of `Y` among `length(x)` bins with centers specified by `x`. For example, if `x` is a 5-element vector, hist distributes the elements of `Y` into five bins centered on the x-axis at the elements in x. Note: use `histc` if it is more natural to specify bin edges instead of centers.

`n = hist(Y,nbins)` where `nbins` is a scalar, uses `nbins` number of bins.

`[n,xout] = hist(...)` returns vectors `n` and `xout` containing the frequency counts and the bin locations. You can use `bar(xout,n)` to plot the histogram.

`hist(...)` without output arguments, hist produces a histogram plot of the output described above. `hist` distributes the bins along the x-axis between the minimum and maximum values of `Y`.

`hist(axes_handle,...)` plots into the axes with handle `axes_handle` instead of the current axes (`gca`).

Remark 3.7.1.
- All elements in vector `Y` or in one column of matrix `Y` are grouped according to their numeric range. Each group is shown as one bin.
- The histogram's x-axis reflects the range of values in `Y`. The histogram's y-axis shows the number of elements that fall within the groups; therefore, the y-axis ranges from 0 to the greatest number of elements deposited in any bin.
- The histogram is created with a patch graphics object. If you want to change the color of the graph, you can set patch properties. See Example 3.7.2 section for more information. By default, the graph color is controlled by the current colormap, which maps the bin color to the first color in the colormap. ♢

Example 3.7.2. Generate a bell-curve histogram from Gaussian data, then change the color of the graph so that the bins are light gray and the edges of the bins are red.

```
x = -2.9:0.1:2.9;
y = randn(10000,1);
hist(y,x)
h = findobj(gca,'Type','patch');
set(h,'FaceColor',[0.7 0.7 0.7],'EdgeColor','r')
```

The graph is given in Figure 3.1 ♢

A rose plot is a histogram created in a polar coordinate system. For example, consider samples of the wind direction taken over a 12-hour period.

```
wdir = [45 90 90 45 360 335 360 270 335 270 335 335];
```

To display this data using the `rose` function, convert the data to radians, then use the data as an argument to the `rose` function. Increase the `LineWidth` property of the line to improve the visibility of the plot (`findobj`).

```
wdir = wdir * pi/180;
rose(wdir)
hline = findobj(gca,'Type','line');
set(hline,'LineWidth',1.5)
```

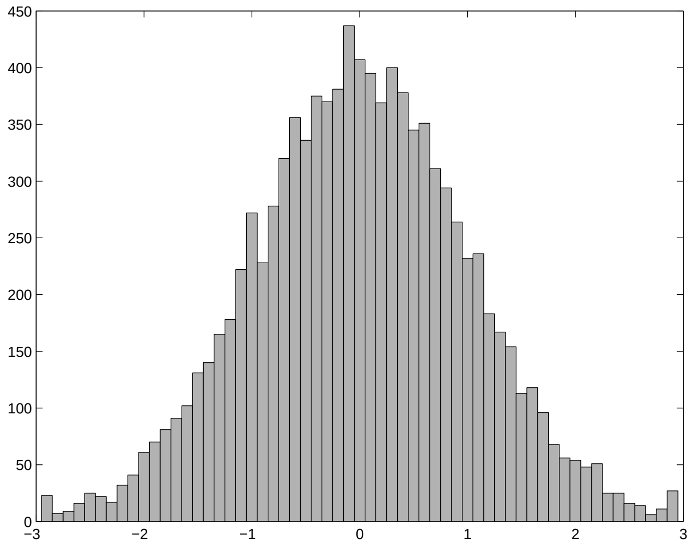


Figure 3.1: A histogram

The plot shows that the wind direction was primarily 335 during the 12-hour period

`n = histc(x,edges)` counts the number of values in vector x that fall between the elements in the edges vector (which must contain monotonically non-decreasing values). `n` is a length(edges) vector containing these counts.

`n(k)` counts the value `x(i)` if `edges(k) > x(i) >= edges(k+1)`. The last bin counts any values of `x` that match `edges(end)`. Values outside the values in edges are not counted. Use `-inf` and `inf` in edges to include all non-NaN values.

For matrices, `histc(x,edges)` returns a matrix of column histogram counts. For N-D arrays, `histc(x,edges)` operates along the first non-singleton dimension.

`n = histc(x,edges,dim)` operates along the dimension dim.

`[n,bin] = histc(...)` also returns an index matrix bin. If `x` is a vector, `n(k) = sum(bin==k)`. `bin` is zero for out of range values. If `x` is an M-by-N matrix, then,

`for j=1:N, n(k,j) = sum(bin(:,j)==k); end`

To plot the histogram, use the `bar` command.

3.7.2 Bar graphs

`bar` and `barh` display the values in a vector or matrix as horizontal or vertical bars.

Example 3.7.3. This example plots a bell-shaped curve as a bar graph and sets the colors of the bars to red.

```
x = -2.9:0.2:2.9;
bar(x,exp(-x.*x),'r')
```

See Figure 3.2. ♢

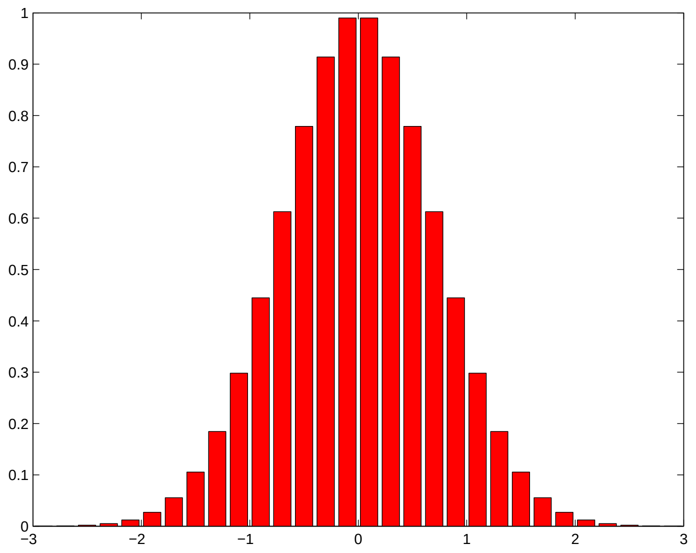


Figure 3.2: A bar plot

Example 3.7.4. This example illustrates some bar graph options.

```
Y = round(rand(5,3)*10);
subplot(2,2,1)
bar(Y,'group')
title 'Group'
subplot(2,2,2)
bar(Y,'stack')
title 'Stack'
subplot(2,2,3)
barh(Y,'stack')
title 'Stack'
subplot(2,2,4)
bar(Y,1.5)
title 'Width = 1.5'
```

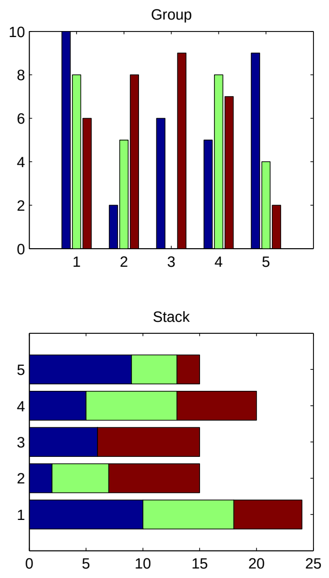


Figure 3.3: Options for bar plot

See Figure 3.3. ♢

`bar3` and `bar3h` draw three-dimensional vertical and horizontal bar charts.

Example 3.7.5. This example creates six subplots showing the effects of different arguments for bar3. The data Y is a seven-by-three matrix generated using the cool colormap:

```
Y = cool(7);
subplot(2,3,1)
bar3(Y,'detached')
title('Detached')
subplot(2,3,2)
bar3(Y,0.25,'detached')
title('Width = 0.25')
subplot(2,3,3)
bar3(Y,'grouped')
title('Grouped')
subplot(2,3,4)
bar3(Y,0.5,'grouped')
title('Width = 0.5')
subplot(2,3,5)
```

```
bar3(Y,'stacked')
title('Stacked')
subplot(2,3,6)
bar3(Y,0.3,'stacked')
title('Width = 0.3')
colormap([1 0 0;0 1 0;0 0 1])
```

See Figure 3.4. ◇

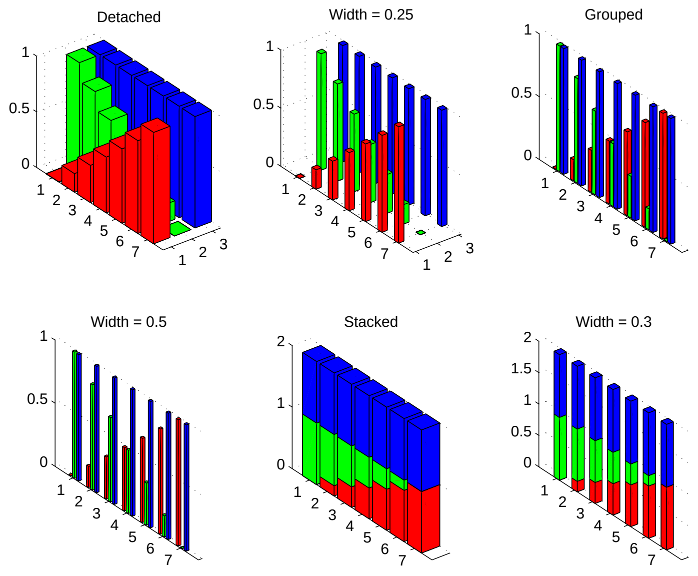


Figure 3.4: A `bar3d` example

3.7.3 Pie charts

pie and pie3 draw bi dimensional and tri-dimensional pie charts respectively.

`pie(X)` draws a pie chart using the data in `X`. Each element in `X` is represented as a slice in the pie chart.

`pie(X,explode)` offsets a slice from the pie. explode is a vector or matrix of zeros and nonzeros that correspond to `X`. A nonzero value offsets the corresponding slice from the center of the pie chart, so that `X(i,j)` is offset from the center if `explode(i,j)` is nonzero. explode must be the same size as `X`.

`pie(...,labels)` specifies text labels for the slices. The number of labels must equal the number of elements in `X`. For example, `pie(1:3,'Taxes','Expenses','Profit')`

`pie(axes_handle,...)` plots into the axes with handle `axes_handle` instead of the current axes `(gca)`.

`h = pie(...)` returns a vector of handles to patch and text graphics objects.

Remark 3.7.6. The values in `X` are normalized via `X/sum(X)` to determine the area of each slice of the pie. If $\mathtt{sum}(\mathtt{X}) \geq 1$, the values in `X` directly specify the area of the pie slices. MATLAB draws only a partial pie if `sum(X)<1`. ♢

Example 3.7.7. Emphasize the second slice in the chart by setting its corresponding explode element to 1.

```
x = [1 3 0.5 2.5 2];
explode = [0 1 0 0 0];
pie(x,explode)
colormap jet
```

See Figure 3.5.

Now a 3D pie chart:

```
x = [1 3 0.5 2.5 2];
explode = [0 1 0 0 0];
pie3(x,explode)
colormap hsv
```

The graph appears in Figure 3.6 ♢

3.7.4 Probability density estimation

You can also describe a data sample by estimating its density in a nonparametric way. The `ksdensity` function does this by using a kernel smoothing function and an associated bandwidth to estimate the density. It uses the so-called "floating-window" method. This method consists of building the class $(x - h/2, x + h/2)$, $h > 0$, finding the frequency f_x of this class, and building the curve given by $\hat{f}(x) = \frac{f_x}{Nh}$, $x \in (a, b)$. A smoothing kernel, $K(x)$, can improves the appealing of this curve, taking

$$\hat{f}(x) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x - x_i'}{h}\right).$$

This smoothing kernel is a symmetric pdf. Example of smoothing kernels:

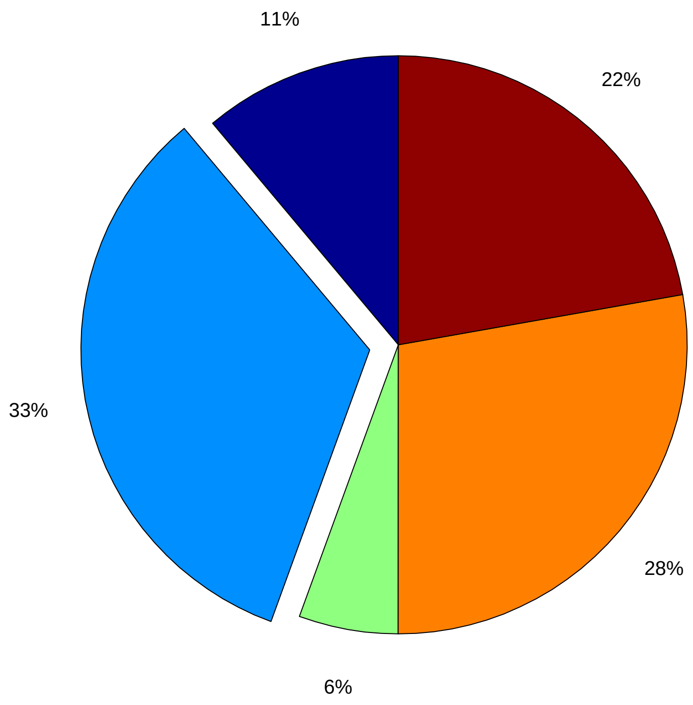


Figure 3.5: A 2D pie chart

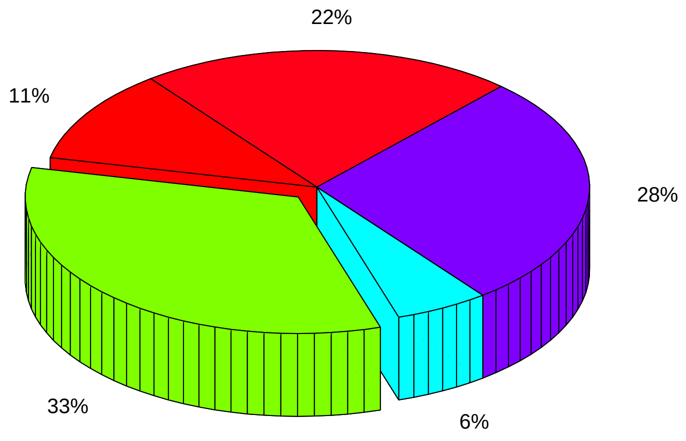


Figure 3.6: A 3D pie chart

- Indicator kernel
$$K(x) = \begin{cases} 1, & \text{if } x \in (-1/2, 1/2); \\ 0, & \text{otherwise.} \end{cases} \tag{3.7.1}$$

- Gaussian kernel
$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \qquad x \in \mathbb{R}. \tag{3.7.2}$$

- parabolic (Epanechnikov) kernel
$$K(x) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1 - \frac{x^2}{5}\right), & \text{if } |x| < \sqrt{5}; \\ 0, & \text{if } |x| \geq \sqrt{5}. \end{cases} \tag{3.7.3}$$

- forth degree
$$K(x) = \begin{cases} \frac{15}{16}(1 - x^2)^2, & \text{if } |x| < 1; \\ 0, & \text{if } |x| \geq 1. \end{cases} \tag{3.7.4}$$

- cosine
$$K(x) = \begin{cases} 1 + \cos(2\pi x), & \text{if } 0 < x < 1; \\ 0, & \text{otherwise.} \end{cases} \tag{3.7.5}$$

- triangular
$$K(x) = \begin{cases} 1 - |x|, & \text{if } |x| < 1; \\ 0, & \text{otherwise.} \end{cases} \tag{3.7.6}$$

Syntax:

```
[f,xi] = ksdensity(x)
f = ksdensity(x,xi)
ksdensity(... )
ksdensity(ax,... )
[f,xi,u] = ksdensity(... )
[... ] = ksdensity(...,'param1',val1,'param2',val2,... )
```

Description
`[f,xi] = ksdensity(x)` computes a probability density estimate of the sample in the vector `x`. `f` is the vector of density values evaluated at the points in `xi`. The estimate is based on a normal kernel function, using a window parameter ('`width`') that is a function of the number of points in `x`. The density is evaluated at 100 equally spaced points covering the range of the data in `x`.

`f = ksdensity(x,xi)` specifies the vector `xi` of values where the density estimate is to be evaluated.
`ksdensity(... )` without output arguments produces a plot of the results.
`ksdensity(ax,... )` plots into axes ax instead of gca.
`[f,xi,u] = ksdensity(... )` also returns the width of the kernel smoothing window.
`[... ] = ksdensity(...,'param1',val1,'param2',val2,... )` specifies parameter name/value pairs to control the density estimation. Valid parameters and their possible values are as follows:

`'censoring'` A logical vector of the same length as `x`, indicating which entries are censoring times. Default is no censoring.

`'kernel'` The type of kernel smoother to use. Choose the value as `'normal'` (default), `'box'`, `'triangle'`, or `'epanechnikov'`. Alternatively, you can specify some other function, as a function handle or as a string, e.g., `@normpdf` or `'normpdf'`. The function must take a single argument that is an array of distances between data values and places where the density is evaluated. It must return an array of the same size containing corresponding values of the kernel function.

`'npoints'` The number of equally-spaced points in `xi`. Default is 100.

`'support'` The support (set where density is nonzero

- **`'unbounded'`** allows the density to extend over the whole real line (default).
- **`'positive'`** restricts the density to positive values.
- **`[lb,up]`** A two-element vector gives finite lower and upper bounds for the support of the density.

`'weights'` Vector of the same length as x, assigning weight to each x value. The default is equal weights).

`'width'` The bandwidth of the kernel smoothing window. The default is optimal for estimating normal densities, but you may want to choose a smaller value to reveal features such as multiple modes.

Example 3.7.8. This example generates a mixture of two normal distributions, and plots the estimated density (see Figure 3.7.

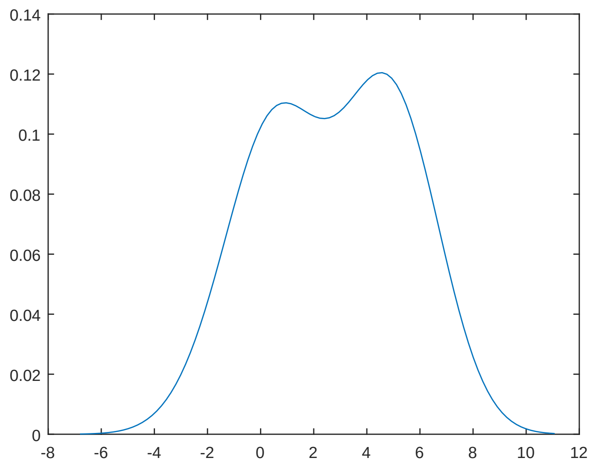


Figure 3.7: A pdf estimation by using kernel smoothing method

```
x = [randn(30,1) ; 5+randn(30,1)] ;
[f,xi] = ksdensity(x) ;
plot(xi,f) ;
```

Try for various kernels and width. ♢

3.7.5 Empirical cumulative distribution function

The `ksdensity` function described in the last section produces an empirical version of a probability density function (pdf). That is, instead of selecting a density with a particular parametric form and estimating the parameters, it produces a nonparametric density estimate that tries to adapt itself to the data.

Similarly, it is possible to produce an empirical version of the cumulative distribution function (cdf). The `ecdf` function computes this empirical cdf. It returns the values of a function F such that $F(x)$ represents the proportion of observations in a sample less than or equal to x.

The idea behind the empirical cdf is simple. It is a function that assigns probability $1/n$ to each of n observations in a sample. Its graph has a stair-step appearance. If a sample comes from a distribution in a parametric family (such as a normal distribution), its empirical cdf is likely to resemble the parametric distribution. If not, its empirical distribution still gives an estimate of the cdf for the distribution that generated the data.

The following example generates 20 observations from a normal distribution with mean 10 and standard deviation 2. You can use `ecdf` to calculate the empirical cdf and stairs to plot it. Then you overlay the normal distribution curve on the empirical function.

```
x = normrnd(10,2,20,1); [f,xf] = ecdf(x);
stairs(xf,f)
xx=linspace(5,15,100);
yy = normcdf(xx,10,2);
hold on; plot(xx,yy,'r:');
hold off
legend('Empirical cdf','Normal cdf',2)
```

The empirical cdf is especially useful in survival analysis applications. In such applications the data may be censored, that is, not observed exactly. Some individuals may fail during a study, and you can observe their failure time exactly. Other individuals may drop out of the study, or may not fail until after the study is complete. The `ecdf` function has arguments for dealing with censored data. In addition, you can use the `coxphfit` function with individuals that have predictors that are not the same.

Another related function is `cdfplot`.

3.8 The Bootstrap

In recent years the statistical literature has examined the properties of resampling as a means to acquire information about the uncertainty of statistical estimators.

The *bootstrap* is a procedure that involves choosing random samples with replacement from a data set and analyzing each sample the same way. Sampling with replacement means that every sample is returned to the data set after sampling. So a particular data point from the original data set could appear multiple times in a given bootstrap sample. The number of elements in each bootstrap sample equals the number of elements in the original data set. The range of sample estimates you obtain enables you to establish the uncertainty of the quantity you are estimating.

Syntax

```
bootstat = bootstrp(nboot,bootfun,d1,d2,...)
[bootstat,bootsam] = bootstrp(...)
```

Description

`bootstat = bootstrp(nboot,bootfun,d1,d2,...)` draws `nboot` bootstrap samples from each of the input data sets, `d1`, `d2`, etc., and passes the bootstrap samples to function `bootfun` for analysis. `bootfun` is a function handle specified using the @ sign. `nboot` must be a positive integer, and each input data set must contain the same number of rows, `n`. Each bootstrap sample contains `n` rows chosen randomly (with replacement) from the corresponding input data set (`d1`, `d2`, etc.). Any scalar arguments among `d1`, `d2`, etc., are passed to `bootfun` unchanged.

Each row of the output `bootstat` contains the results of applying `bootfun` to one bootstrap sample. If `bootfun` returns multiple output arguments, only the first is stored in `bootstat`. If the first output from `bootfun` is a matrix, the matrix is reshaped to a row vector for storage in `bootstat`.
`[bootstat,bootsam] = bootstrp(...  )` returns an n-by-n boot matrix of bootstrap indices, `bootsam`. Each column in `bootsam` contains indices of the values that were drawn from the original data sets to constitute the corresponding bootstrap sample. For example, if `d1`, `d2`, etc., each contain 16 values, and `nboot` = 4, then `bootsam` is a 16-by-4 matrix. The first column contains the indices of the 16 values drawn from `d1`, `d2`, etc., for the first of the four bootstrap samples, the second column contains the indices for the second of the four bootstrap samples, and so on. (The bootstrap indices are the same for all input data sets.) To get the output samples `bootsam` without applying a function, set `bootfun` to empty ([]).

3.8.1 Examples

Estimating the Density of Bootstrapped Statistic

Compute a sample of 100 bootstrapped means of random samples taken from the vector Y, and plot an estimate of the density of these bootstrapped means:

```
y = exprnd(5,100,1) ;
m = bootstrp(100, @mean, y) ;
[fi,xi] = ksdensity(m) ;
plot(xi,fi) ;
```

Bootstrapping More Than One Statistic

Compute a sample of 100 bootstrapped means and standard deviations of random samples taken from the vector Y, and plot the bootstrap estimate pairs:

```
y = exprnd(5,100,1);
stats = bootstrp(100, @(x)[mean(x) std(x)],y);
plot(stats(:,1),stats(:,2),'o')
```

Bootstrapping a Regression Model

Estimate the standard errors for a coefficient vector in a linear regression by bootstrapping residuals:

```
load hald ingredients heat
x = [ones(size(heat)),ingredients];
y = heat;
b = regress(y,x);
yfit = x*b;
resid = y-yfit;
se=std(bootstrp(1000,@(bootr)regress(yfit+bootr,x),resid));
```

3.9 Statistical plots

The Statistics Toolbox adds specialized plots to the extensive graphics capabilities of MATLAB:

- Box plots are graphs for describing data samples. They are also useful for graphic comparisons of the means of many samples (see One-Way Analysis of Variance (ANOVA) on page 4-4).
- Distribution plots are graphs for visualizing the distribution of one or more samples. They include normal and Weibull probability plots, quantile-quantile plots, and empirical cumulative distribution plots.
- Scatter plots are graphs for visualizing the relationship between a pair of variables or several such pairs. Grouped versions of these plots use different plotting symbols to indicate group membership. The gname function can label points on these plots with a text label or an observation number.

3.9.1 Boxplots

The graph in figure 3.8 shows an example of a notched box plot.

This plot has several graphic elements:

- The lower and upper lines of the box are the 25th and 75th percentiles of the sample. The distance between the top and bottom of the box is the interquartile range.

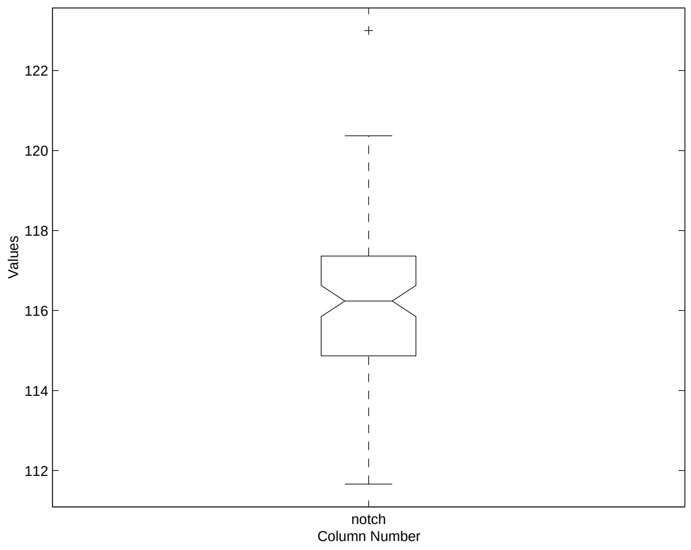


Figure 3.8: A typical boxplot

- The line in the middle of the box is the sample median. If the median is not centered in the box, that is an indication of skewness.
- The whiskers are lines extending above and below the box. They show the extent of the rest of the sample (unless there are outliers). Assuming no outliers, the maximum of the sample is the top of the upper whisker. The minimum of the sample is the bottom of the lower whisker. By default, an outlier is a value that is more than 1.5 times the interquartile range away from the top or bottom of the box.
- The plus sign at the top of the plot is an indication of an outlier in the data. This point might be the result of a data entry error, a poor measurement, or a change in the system that generated the data.
- The notches in the box are a graphic confidence interval about the median of a sample. Box plots do not have notches by default.

In its most general form

```
boxplot(X,notch,'sym',vert,whis)
```

`boxplot` allows you to specify if graph has a notch (`notch=1`) or no (`notch`=0), a symbol `sym` to indicate any outlier, if there exists, if box is horizontal (`vert`=0) or vertical (`vert`=1) and the length of the ”whiskers.” `whis` defines the maximum length of the whiskers as a function of the inter-quartile range (default = 1.5 * IQR). Each whisker extends to the most extreme data value within whis*IQR of the box. If `whis` = 0, then `boxplot` displays all data values outside the box using the plotting symbol, '`sym`'.

A side-by-side comparison of two notched box plots provides a graphical way to determine which groups have significantly different medians. This is similar to a one-way analysis of variance, except that the latter compares means. Analysis of variance is described in Chapter Linear Models.

Example 3.9.1. This example compares two sequences of 100 normal random numbers.

```
x1 = normrnd(5,1,100,1);
x2 = normrnd(6,1,100,1);
x = [x1 x2];
boxplot(x,1)
```

Figure 3.9 gives the graph. The difference between the means of the two columns of x is 1. We can detect this difference graphically by observing that the notches in the boxplot do not overlap. ♢

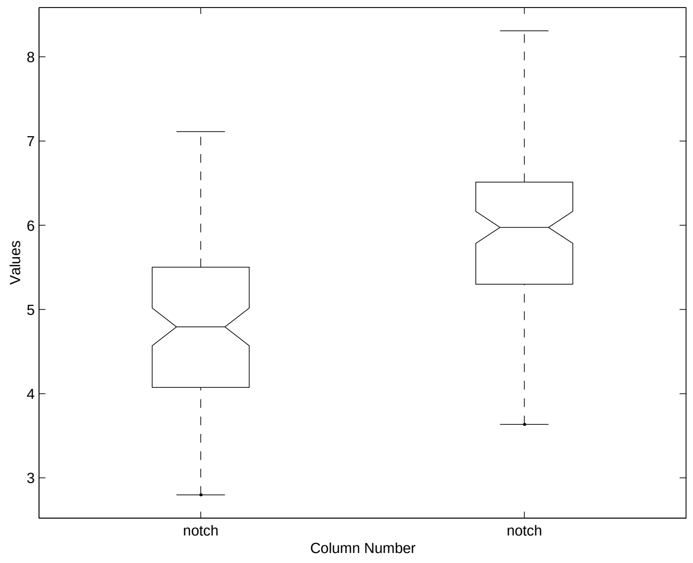


Figure 3.9: Compare two samples using `boxplot`

3.9.2 Distribution plots

There are several types of plots for examining the distribution of one or more samples:

- Normal Probability Plots;
- Quantile-Quantile Plots
- Weibull Probability Plots
- Empirical Cumulative Distribution Function (CDF)

Normal probability plots

A normal probability plot is a useful graph for assessing whether data comes from a normal distribution. Many statistical procedures make the assumption that the underlying distribution of the data is normal, so this plot can provide some assurance that the assumption of normality is not being violated, or provide an early warning of a problem with your assumptions.

This example shows a typical normal probability plot (Figure 3.10).

```
x = normrnd(10,1,25,1);
normplot(x)
```

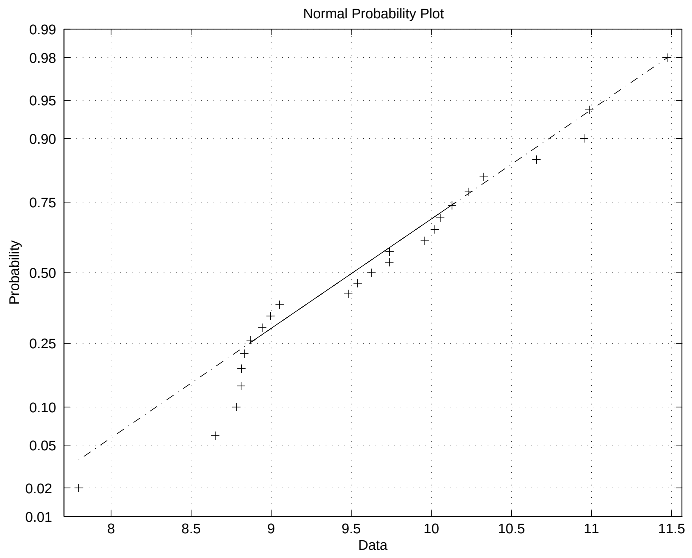


Figure 3.10: A typical normal probability plot

The plot has three graphical elements. The plus signs show the empirical probability versus the data value for each point in the sample. The solid line connects the 25th and 75th percentiles of the data and represents a robust linear fit (i.e., insensitive to the extremes of the sample). The dashed line extends the solid line to the ends of the sample.

The scale of the y-axis is not uniform. The y-axis values are probabilities and, as such, go from zero to one. The distance between the tick marks on the y-axis matches the distance between the quantiles of a normal distribution. The quantiles are close together near the median (probability = 0.5) and stretch out symmetrically moving away from the median. Compare the vertical distance from the bottom of the plot to the probability 0.25 with the distance from 0.25 to 0.50. Similarly, compare the distance from the top of the plot to the probability 0.75 with the distance from 0.75 to 0.50.

If all the data points fall near the line, the assumption of normality is reasonable. But, if the data is nonnormal, the plus signs may follow a curve, as in the example using exponential data below (Figure 3.11).

```
x = exprnd(10,100,1);
normplot(x)
```

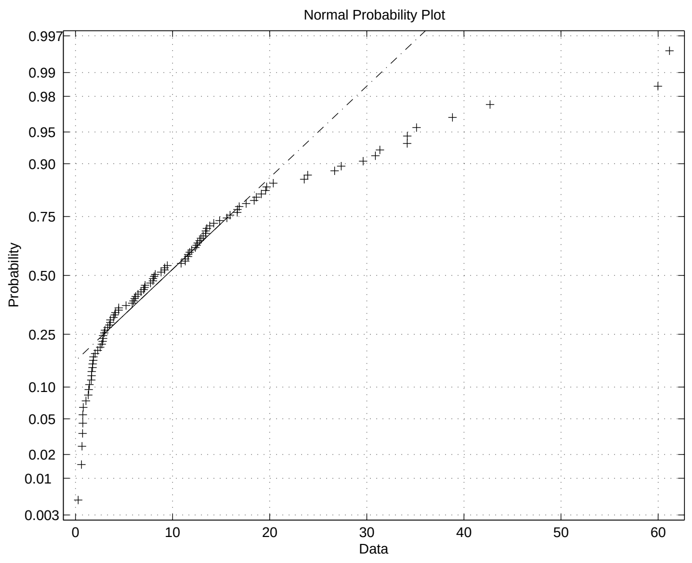


Figure 3.11: A normal probability plot for a nonnormal distribution

This plot is clear evidence that the underlying distribution is not normal.

Quantile-quantile plots

A quantile-quantile plot is useful for determining whether two samples come from the same distribution (whether normally distributed or not).

The example shows a quantile-quantile plot of two samples from a Poisson distribution (Figure 3.12).

```
x = poissrnd(10,50,1);
y = poissrnd(5,100,1);
qqplot(x,y);
```

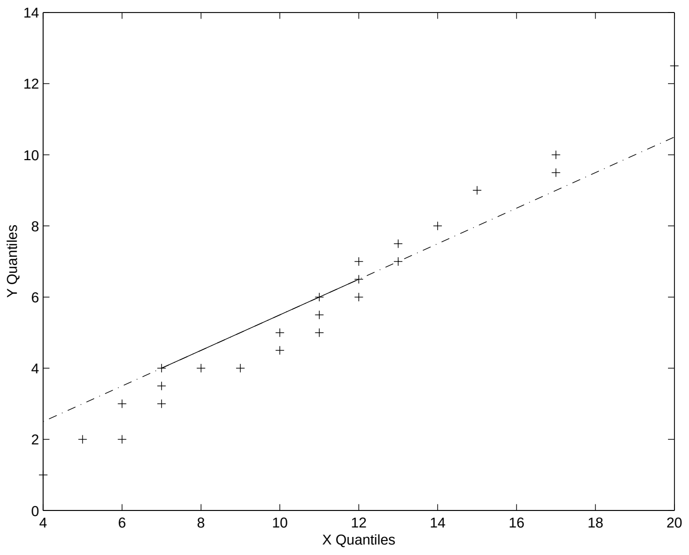


Figure 3.12: A quantile-quantile plot

Even though the parameters and sample sizes are different, the straight line relationship shows that the two samples come from the same distribution.

Like the normal probability plot, the quantile-quantile plot has three graphical elements. The pluses are the quantiles of each sample. By default the number of pluses is the number of data values in the smaller sample. The solid line joins the 25th and 75th percentiles of the samples. The dashed line extends the solid line to the extent of the sample.

The example below shows what happens when the underlying distributions are not the same (Figure 3.13).

```
x = normrnd(5,1,100,1);
y = weibrnd(2,0.5,100,1);
qqplot(x,y);
```

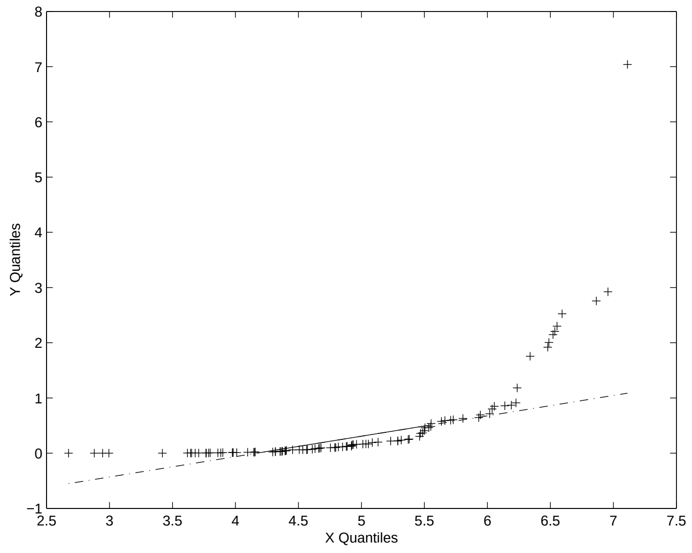


Figure 3.13: A quantile-quantile plot for samples having different distributions

These samples clearly are not from the same distribution.

It is incorrect to interpret a linear plot as a guarantee that the two samples come from the same distribution. But, for assessing the validity of a statistical procedure that depends on the two samples coming from the same distribution (e.g., ANOVA), a linear quantile-quantile plot should be sufficient.

Weibull probability plots

A Weibull probability plot is a useful graph for assessing whether data comes from a Weibull distribution. Many reliability analyses make the assumption that the underlying distribution of the lifetimes is Weibull, so this plot can provide some assurance that this assumption is not being violated, or provide an early warning of a problem with your assumptions.

The scale of the y-axis is not uniform. The y-axis values are probabilities and, as such, go from zero to one. The distance between the tick marks on the y-axis matches the distance between the quantiles of a Weibull distribution.

If the data points (pluses) fall near the line, the assumption that the data comes from a Weibull distribution is reasonable.

This example shows a typical Weibull probability plot (Figure 3.14).

```
y = weibrnd(2,0.5,100,1);
```

```
weibplot(y)
```

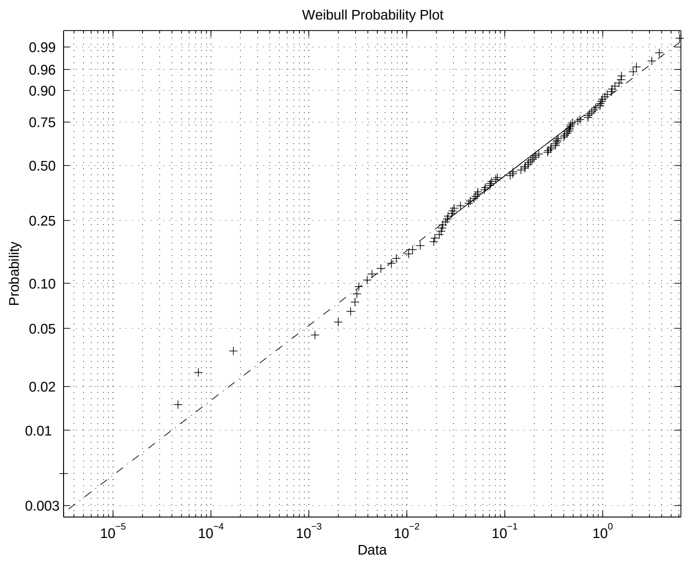


Figure 3.14: A Weibull probability plot

Empirical cumulative distribution function

If you are not willing to assume that your data follows a specific probability distribution, you can use the `cdfplot` function to graph an empirical estimate of the cumulative distribution function (cdf). This function computes the proportion of data points less than each `x` value, and plots the proportion as a function of `x`. The `y`-axis scale is linear, not a probability scale for a specific distribution.

This example shows the empirical cumulative distribution function for a Weibull sample (Figure 3.15).

```
y = weibrnd(2,0.5,100,1);
cdfplot(y)
```

The plot shows a probability function that rises steeply near `x=0` and levels off for larger values. Over 80% of the observations are less than 1, with the remaining values spread over the range [1 5].

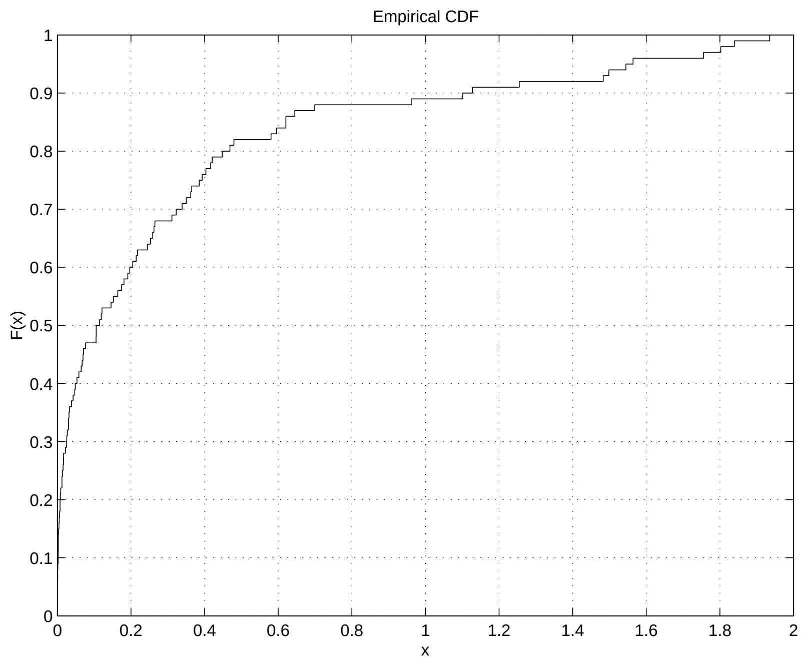


Figure 3.15: An empirical cdf plot

3.9.3 Scatter plots

A scatter plot is a simple plot of one variable against another. The MATLAB `plot` and `scatter` functions can produce scatter plots. The MATLAB `plotmatrix` function can produce a matrix of such plots showing the relationship between several pairs of variables.

The Statistics Toolbox adds functions that produce grouped versions of these plots. These are useful for determining whether the values of two variables or the relationship between those variables is the same in each group.

Suppose you want to examine the weight and mileage of cars from three different model years (Figure 3.16.

```
load carsmall
gscatter(Weight,MPG,Model_Year,'','xos')
```

This shows that not only is there a strong relationship between the weight of a car and its mileage, but also that newer cars tend to be lighter and have better gas mileage than older cars.

(The default arguments for `gscatter` produce a scatter plot with the different groups shown with the same symbol but different colors. The last two arguments above request that all groups be shown in default colors and with different symbols.)

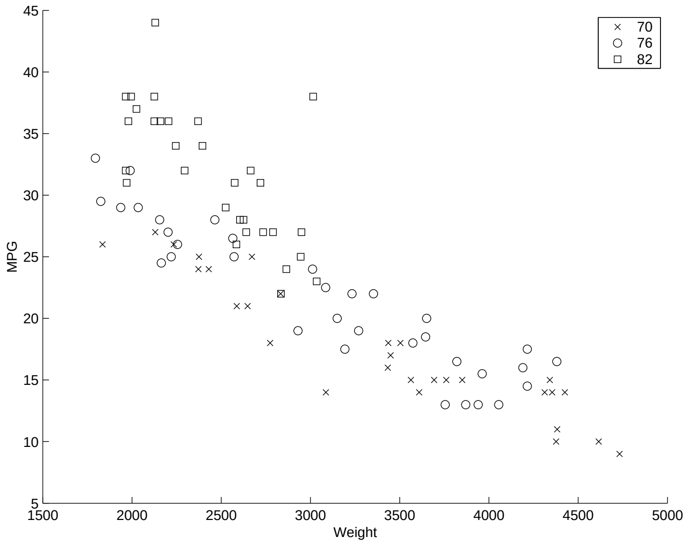


Figure 3.16: A `gscatter` example

The `carsmall` data set contains other variables that describe different aspects of cars. You can examine several of them in a single display by creating a grouped plot matrix (Figure 3.17).

```
xvars = [Weight Displacement Horsepower];
yvars = [MPG Acceleration];
gplotmatrix(xvars,yvars,Model_Year,'','xos')
```

The upper right subplot displays `MPG` against `Horsepower`, and shows that over the years the horsepower of the cars has decreased but the gas mileage has improved.

The `gplotmatrix` function can also graph all pairs from a single list of variables, along with histograms for each variable. See Multivariate Analysis of Variance (MANOVA).

```
function [yCDF,xCDF,n] = mycdfcalc(x)
%MYCDFCALC Calculate an empirical cdf.
%   [YCDF,XCDF] = CDFCALC(X) calculates an empirical
%   cumulative distribution function (CDF) of the
%   observations in the data sample vector X.
%   X may be a row or column vector, and represents
%   a random sample of observations from some underlying
%   distribution.  On return XCDF is the set of X values
```

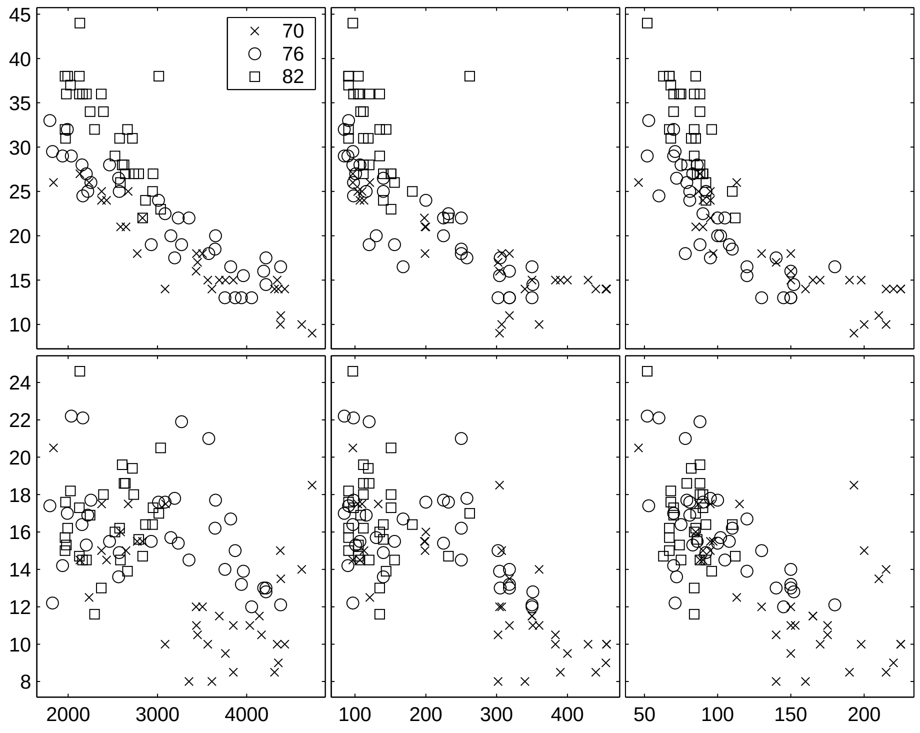


Figure 3.17: A `gplotmatrix` example

```
%   at which the CDF increases. At XCDF(i), the function
%   increases from YCDF(i) to YCDF(i+1).


% Remove missing observations indicated by NaN’s.
x = x(˜isnan(x));
n = length(x);

% Sort observation data in ascending order.
x = sort(x(:));

%
% Compute cumulative sum such that the sample CDF is
% F(x) = (number of observations <= x)/
% (total number of observations).
% Note that the bin edges are padded with +/- infinity
% for auto-scaling of the x-axis.
%

% Get cumulative sums
yCDF = (1:n)’ / n;

% Remove duplicates; only need final one with total count
notdup = ([diff(x(:)); 1] > 0);
xCDF = x(notdup);
yCDF = [0; yCDF(notdup)];
```

```
function [handleCDF] = mycdfplot(x)
%CDFPLOT Display an empirical cdf.
%   CDFPLOT(X) plots an empirical cumulative
%   distribution function (CDF) of the observations
%   in the data sample vector X. X may be a row or
%   column vector, and represents a random sample
%   of observations from  some underlying distribution.
%   H = CDFPLOT(X) plots F(x), returns a handle
%

% Get sample cdf, display error message if any
[yy,xx,n] = mycdfcalc(x);
```

```
% Create vectors for plotting
k = length(xx);
n = reshape(repmat(1:k, 2, 1), 2*k, 1);
xCDF    = [-Inf; xx(n); Inf];
yCDF    = [0; 0; yy(1+n)];

%
% Now plot the sample (empirical) CDF staircase.
%

hCDF = plot(xCDF , yCDF);
if (nargout>0), handleCDF=hCDF; end
grid  (’on’)
xlabel(’x’)
ylabel(’F(x)’)
title (’Empirical CDF’)
```

Chapter 4

Estimation

4.1 Mean and Variance as a Function of Parameters

The mean and variance of a probability distribution are generally simple functions of the parameters of the distribution. The Statistics Toolbox functions ending in ’stat’ all produce the mean and variance of the desired distribution for the given parameters.

The example below shows a contour plot of the mean of the Weibull distribution as a function of the parameters.

```
x = (0.5:0.1:5) ;
y = (1:0.04:2) ;
[X,Y] = meshgrid(x,y) ;
Z = weibstat(X,Y) ;
[c,h] = contour(x,y,Z,[0.4 0.6 1.0 1.8]) ;
clabel(c) ;
```

The graph is give in Figure 4.1.

4.2 Parameter Estimation

The `mle` function computes the maximum likelihood estimation and confidence intervals for distribution parameters.

Syntax:

```
phat = mle(’dist’,data)
[phat,pci] = mle(’dist’,data)
[phat,pci] = mle(’dist’,data,alpha)
```

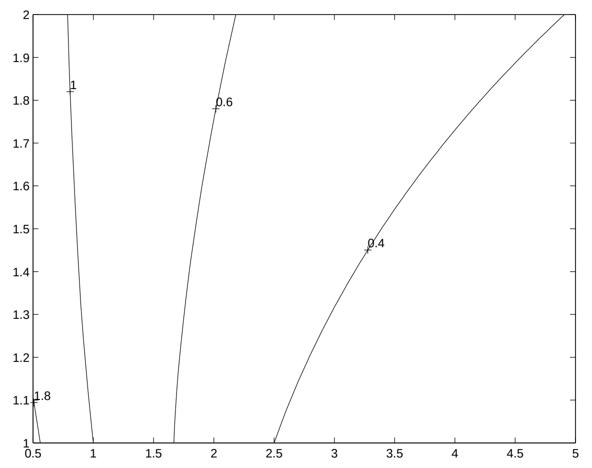


Figure 4.1: Contour plot for means of various Weibull distributions

```
[phat,pci] = mle('dist',data,alpha,p1)
```

Description

`phat = mle('dist',data)` returns the maximum likelihood estimates (MLEs) for the distribution specified in '`dist`' using the sample in the vector, `data`. See Overview of the Distributions for the list of available distributions.

`[phat,pci] = mle('dist',data)` returns the MLEs and 95% percent confidence intervals.

`[phat,pci] = mle('dist',data,alpha)` returns the MLEs and 100 * (1 - `alpha`)% confidence intervals given the `data` and the specified `alpha`.

`[phat,pci] = mle('dist',data,alpha,p1)` is used for the binomial distribution only, where `p1` is the number of trials.

Example

```
rv = binornd(20,0.75)
rv =
    16
[p,pci] = mle('binomial',rv,0.05,20)
p =
    0.8000
pci =
    0.5634
    0.9427
```

`betafit`	Parameter estimates and confidence intervals for beta distributed data
`betalike`	Negative beta log-likelihood function
`binofit`	Parameter estimates and confidence intervals for binomial data
`evfit`	Parameter estimates and confidence intervals for extreme value data
`evlike`	Negative log-likelihood for the extreme value distribution
`expfit`	Parameter estimates and confidence intervals for exponential data
`explike`	Negative log-likelihood for the exponential distribution
`gamfit`	Parameter estimates and confidence intervals for gamma distributed data
`gamlike`	Negative gamma log-likelihood function
`gevlike`	Negative log-likelihood for the generalized extreme value distribution
`gplike`	Negative log-likelihood for the generalized Pareto distribution
`lognfit`	Parameter estimates and confidence intervals for lognormal data
`lognlike`	Negative log-likelihood for the lognormal distribution
`nbinfit`	Parameter estimates and confidence intervals for negative binomial data
`normfit`	Parameter estimates and confidence intervals for normal data
`normlike`	Negative normal log-likelihood function
`poissfit`	Parameter estimates and confidence intervals for Poisson data
`raylfit`	Parameter estimates and confidence intervals for Rayleigh data
`unifit`	Parameter estimates for uniformly distributed data
`wblfit`	Parameter estimates and confidence intervals for Weibull data
`wbllike`	Weibull negative log-likelihood function

Functions ended in -like returns the negative log-likelihood function. Their calls have one of the forms

```
logL = FunctionName(params,data)
[logL,avar] = FunctionName(params,data)
```

`FunctionName` is the function name, `params` is the list of parameters, and `data` represents the sample data. The output parameter `logL` is the negative log-likelihood function and `avar` is the inverse of Fisher's information matrix. The diagonal elements of `avar` are the asymptotic variances of their respective parameters. Since this functions returns the negative log-likelihood function, minimizing this function using `fminsearch` is the same as maximizing the likelihood.

Functions ended in -fit returns parameter point estimates and confidence intervals for parameters of the distribution.

4.3 Examples

Example 4.3.1 (Parameter estimation for the beta distribution). Suppose you are collecting data that has hard lower and upper bounds of zero and one respectively. The function `betafit` returns the MLEs and confidence intervals for the parameters of the beta distribution. Here is an example using random numbers from the beta distribution with `a=5` and `b=0.2`.

```
>> r = betarnd(5,0.2,100,1) ;
>> [phat, pci] = betafit(r)
phat =
    5.6026    0.2192
pci =
    3.6172    0.1587
    7.5880    0.2797
```

The MLE for parameter `a` is 4.5330, compared to the true value of 5. The 95% confidence interval for `a` goes from 2.8051 to 6.2610, which includes the true value. Similarly the MLE for parameter `b` is 0.2301, compared to the true value of 0.2. The 95% confidence interval for `b` goes from 0.1771 to 0.2832, which also includes the true value. In this made-up example you know the “true value”. In experimentation you do not. ♢

Example 4.3.2 (Parameter estimation for the binomial distribution). Suppose you are collecting data from a widget manufacturing process, and you record the number of widgets within specification in each batch of 100. You might be interested in the probability that an individual widget is within specification. Parameter estimation is the process of determining the parameter, `p`, of the binomial distribution that fits this data best in some sense.

The function binofit returns the MLEs and confidence intervals for the parameters of the binomial distribution. Here is an example using random numbers from the binomial distribution with `n=100` and `p=0.9`.

```
>> r = binornd(100,0.9)
r =
    88
>> [phat, pci] = binofit(r,100)
phat =
    0.8800
pci =
    0.7998    0.9364
```

The MLE for parameter `p` is 0.8800, compared to the true value of 0.9. The 95% confidence interval for `p` goes from 0.7998 to 0.9364, which includes the true value. In this made-up example you know the “true value” of `p`. In experimentation you do not. $\diamondsuit$

Example 4.3.3 (Parameter estimation for the exponential distribution). Suppose you are stress testing light bulbs and collecting data on their lifetimes. You assume that these lifetimes follow an exponential distribution. You want to know how long you can expect the average light bulb to last. Parameter estimation is the process of determining the parameters of the exponential distribution that fit this data best in some sense.

The function `expfit` returns the MLEs and confidence intervals for the parameters of the exponential distribution. Here is an example using random numbers from the exponential distribution with $\mu = 700$.

```
>> lifetimes = exprnd(700,100,1);
>> [muhat, muci] = expfit(lifetimes)
muhat =
  773.8507
muci =
  629.6359
  932.7142
```

The MLE for parameter μ is 672, compared to the true value of 700. The 95% confidence interval for goes from 547 to 811, which includes the true value. In the life tests you do not know the true value of μ so it is nice to have a confidence interval on the parameter to give a range of likely values. $\diamondsuit$

Example 4.3.4 (Parameter Estimation for the Extreme Value Distribution). The function `evfit` returns the maximum likelihood estimates (MLEs) and confidence intervals for the parameters of the extreme value distribution. The following example shows how to fit some sample data using `evfit`, including estimates of the mean and variance from the fitted distribution.

Suppose you want to model the size of the smallest washer in each batch of 1000 from a manufacturing process. If you believe that the sizes are independent within and between each batch, you can fit an extreme value distribution to measurements of the minimum diameter from a series of eight experimental batches. The following code returns the MLEs of the distribution parameters as `parmhat` and the confidence intervals as the columns of `parmci`.

```
x =[19.774 20.141 19.44 20.511 21.377 19.003 19.66 18.83];
[parmhat, parmci] = evfit(x)
```

You can find mean and variance of the extreme value distribution with these parameters using the function `evstat`.

```
[meanfit, varfit] = evstat(parmhat(1),parmhat(2))
meanfit =
   19.776
varfit =
   1.1123
```

Example 4.3.5 (Parameter estimation for the gamma distribution). Suppose you are stress testing computer memory chips and collecting data on their lifetimes. You assume that these lifetimes follow a gamma distribution. You want to know how long you can expect the average computer memory chip to last.

The function `gamfit` returns the MLEs and confidence intervals for the parameters of the gamma distribution. Here is an example using random numbers from the gamma distribution with `a=10` and `b=5`.

```
>> lifetimes = gamrnd(10,5,100,1);
>> [phat, pci] = gamfit(lifetimes)
phat =
    9.6597    5.3535
pci =
    6.9900    3.9070
   12.3294    6.8001
```

Note `phat(1)`=$\hat{a}$ and `phat(2)`=$\hat{b}$. The MLE for parameter a is 9.6597, compared to the true value of 10. The 95% confidence interval for a goes from 6.99 to 12.3294, which includes the true value.

Similarly the MLE for parameter b is 5.3535, compared to the true value of 5. The 95% confidence interval for b goes from 3.9 to 6.8, which also includes the true value.

In the life tests you do not know the true value of a and b so it is nice to have a confidence interval on the parameters to give a range of likely values. $\diamondsuit$

Example 4.3.6 (Parameter Estimation for the Generalized Extreme Value Distribution). If you generate 250 blocks of 1000 random values drawn from Students t distribution with 5 degrees of freedom, and take their maxima, you can fit a generalized extreme value distribution to those maxima. If you generate 250 blocks of 1000 random values drawn from Students t distribution with 5 degrees of freedom, and take their maxima, you can fit a generalized extreme value distribution to those maxima.

```
blocksize = 1000;
nblocks = 250;
t = trnd(5,blocksize,nblocks);
x = max(t); %250 column maxima
paramEsts = gevfit(x)
paramEsts =
   0.1507
   1.2712
   5.8816
```

Notice that the shape parameter estimate (the first element) is positive, which is what you would expect based on block maxima from a Students t distribution.

```
hist(x,2:20);
set(get(gca,’child’),’FaceColor’,[.9 .9 .9])
xgrid = linspace(2,20,1000);
line(xgrid,nblocks*gevpdf(xgrid,paramEsts(1),paramEsts(2),...
     paramEsts(3)));
```

Example 4.3.7 (Parameter estimation for the generalized Pareto distribution). If you generate a large number of random values from a Students t distribution with 5 degrees of freedom, and then discard everything less than 2, you can fit a generalized Pareto distribution to those exceedences.

```
t = trnd(5,5000,1);
y = t(t > 2)-2;
paramEsts = gpfit(y)
paramEsts =
  0.1598
  0.7968
```

Notice that the shape parameter estimate (the first element) is positive, which is what you would expect based on exceedences from a Students t distribution.

```
hist(y+2,2.25:.5:11.75);
set(get(gca,’child’),’FaceColor’,[.9 .9 .9])
xgrid = linspace(2,12,1000);
line(xgrid,.5*length(y)*gppdf(xgrid,paramEsts(1),...
   paramEsts(2),2));
```

Example 4.3.8 (Parameter Estimation for the Negative Binomial Distribution). Suppose you are collecting data on the number of auto accidents on a busy highway, and would like to be able to model the number of accidents per day. Because these are count data, and because there are a very large number of cars and a small probability of an accident for any specific car, you might think to use the Poisson distribution. However, the probability of having an accident is likely to vary from day to day as the weather and amount of traffic change, and so the assumptions needed for the Poisson distribution are not met. In particular, the variance of this type of count data sometimes exceeds the mean by a large amount. The data below exhibit this effect: most days have few or no accidents, and a few days have a large number.

```
>> accident=[2 3 4 2 3 1 12 8 14 31 23 1 10 7 0];
>> mean(accident)
ans =
    8.0667
>> var(accident)
ans =
   79.3524
```

The negative binomial distribution is more general than the Poisson, and is often suitable for count data when the Poisson is not. The function `nbinfit` returns the maximum likelihood estimates (MLEs) and confidence intervals for the parameters of the negative binomial distribution. Here are the results from fitting the accident data:

```
>> [phat,pci] = nbinfit(accident)
phat =
    1.0060    0.1109
pci =
    0.2152    0.0171
    1.7968    0.2046
```

It is difficult to give a physical interpretation in this case to the individual parameters. However, the estimated parameters can be used in a model for the number of daily accidents. For example, a plot of the estimated cumulative probability function shows that while there is an estimated 10% chance of no accidents on a given day, there is also about a 10% chance that there will be 20 or more accidents.

```
plot(0:50,nbincdf(0:50,phat(1),phat(2)),'.-');
xlabel('Accidents per Day')
ylabel('Cumulative Probability')
```

Example 4.3.9 (Parameter Estimation for the Normal Distribution). To use statistical parameters such as mean and standard deviation reliably, you need to have a good estimator for them. The maximum likelihood estimates (MLEs) provide one such estimator. However, an MLE might be biased, which means that its expected value of the parameter might not equal the parameter being estimated. For example, an MLE is biased for estimating the variance of a normal distribution. An unbiased estimator that is commonly used to estimate the parameters of the normal distribution is the minimum variance unbiased estimator (MVUE). The MVUE has the minimum variance of all unbiased estimators of a parameter.

The MVUEs of parameters μ and σ^2 for the normal distribution are the sample average and variance. The sample average is also the MLE for μ. The following are two common formulas for the variance.

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2, \tag{4.3.1}$$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2, \tag{4.3.2}$$

where

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

$\diamondsuit$

Equation (4.3.2) is the maximum likelihood estimator for σ^2, and equation (4.3.2) is the MVUE.

As an example, suppose you want to estimate the mean, μ, and the variance, σ^2, of the heights of all 4th grade children in the United States. The function `normfit` returns the MVUE for μ, the square root of the MVUE for s2, and confidence intervals for and s2. Here is a playful example modeling the heights in inches of a randomly chosen 4th grade class.

```
>> height = normrnd(50,2,30,1); %Simulate heights.
>>[mu,s,muci,sci] = normfit(height)
mu =
   50.1717
s =
    1.9448
muci =
   49.4455
   50.8979
```

```
sci =
    1.5489
    2.6144
```

Note that $\sigma^2$is the MVUE of the variance.

```
>> sˆ2
ans =
    3.7823
```

Example 4.3.10 (Parameter estimation for the Poisson distribution). The MLE and the MVUE of the Poisson parameter, λ, is the sample mean. The sum of independent Poisson random variables is also Poisson distributed with the parameter equal to the sum of the individual parameters. The Statistics Toolbox makes use of this fact to calculate confidence intervals for λ. As λ gets large the Poisson distribution can be approximated by a normal distribution with $\mu = \lambda$ and $\sigma^2 = \lambda$. The Statistics Toolbox uses this approximation for calculating confidence intervals for values of λ greater than 100.

```
>> r = poissrnd(5,10,2);
>> [l,lci] = poissfit(r)
l =
    4.2000    6.5000
lci =
    3.0270    5.0166
    5.6772    8.2848
```

Example 4.3.11 (Parameter estimation for the Rayleigh Distribution). The `raylfit` function returns the MLE of the Rayleigh parameter. This estimate is

$$b = \sqrt{\frac{1}{2n}\sum_{i=1}^{n} x_i^2}.$$

```
>> a=raylrnd(2,100,1);
>> [bhat, bci]=raylfit(a)
bhat =
    1.8271
bci =
    1.6642
    2.0255
```

The estimated value is 1.8271; the confidence interval contains the real value, 2. ♢

Example 4.3.12 (Parameter Estimation for the Uniform Distribution). The sample minimum and maximum are the MLEs of a and b respectively.

```
>> x=unifrnd(0,1,100,1);
>> [min(x),max(x)]
ans =
    0.0038    0.9997
>> [ahat,bhat,ACI,BCI] = unifit(x)
ahat =
    0.0038
bhat =
    0.9997
ACI =
   -0.0265
    0.0038
BCI =
    0.9997
    1.0300
```

Example 4.3.13 (Parameter estimation for the Weibull distribution). Suppose you want to model the tensile strength of a thin filament using the Weibull distribution. The function `wblfit` gives maximum likelihood estimates and confidence intervals for the Weibull parameters.

```
>> strength = weibrnd(0.5,2,100,1); %Simulated strengths.
>> [p,ci] = weibfit(strength)
p =
    0.4539    2.1408
ci =
    0.3281    1.7982
    0.5798    2.4834
```

4.4 Distribution Fitting Tool

Distribution fitting tool is a graphical user interface (GUI) for fitting univariate distribution data. It is a complex GUI with a lot of capabilities and well-designed facilities. To run it type `dfittool`

Chapter 5

Hypothesis Tests

5.1 Introduction

A hypothesis test is a procedure for determining if an assertion about a characteristic of a population is reasonable.

For example, suppose that someone says that the average price of a gallon of regular unleaded gas in Massachusetts is \$1.15. How would you decide whether this statement is true? You could try to find out what every gas station in the state was charging and how many gallons they were selling at that price. That approach might be definitive, but it could end up costing more than the information is worth.

A simpler approach is to find out the price of gas at a small number of randomly chosen stations around the state and compare the average price to \$1.15.

Of course, the average price you get will probably not be exactly \$1.15 due to variability in price from one station to the next. Suppose your average price was \$1.18. Is this three cent difference a result of chance variability, or is the original assertion incorrect? A hypothesis test can provide an answer.

5.2 Hypothesis Test Terminology

To get started, there are some terms to define and assumptions to make:

- The *null hypothesis* is the original assertion. In this case the null hypothesis is that the average price of a gallon of gas is \$1.15. The notation is $H_0 : \mu = 1.15$.

- There are three possibilities for the *alternative hypothesis*. You might only be interested in the result if gas prices were actually higher. In this case, the alternative hypothesis is $H_1 : \mu > 1.15$. The other possibilities are $H_1 : \mu < 1.15$ and $H_1 : \mu \neq 1.15$.

- The *significance level* is related to the degree of certainty you require in order to reject the null hypothesis in favor of the alternative. By taking a small sample you cannot be certain about your conclusion. So you decide in advance to reject the null hypothesis if the probability of observing your sampled result is less than the significance level. For a typical significance level of 5%, the notation is $\alpha = 0.05$. For this significance level, the probability of incorrectly rejecting the null hypothesis when it is actually true is 5%. If you need more protection from this error, then choose a lower value of α.

- The *p-value* is the probability of observing the given sample result under the assumption that the null hypothesis is true. If the p-value is less than α, then you reject the null hypothesis. For example, if $\alpha = 0.05$ and the p-value is 0.03, then you reject the null hypothesis. The converse is not true. If the p-value is greater than a, you have insufficient evidence to reject the null hypothesis.

- The outputs for many hypothesis test functions also include *confidence intervals*. Loosely speaking, a confidence interval is a range of values that have a chosen probability of containing the true hypothesized quantity. Suppose, in the example, 1.15 is inside a 95% confidence interval for the mean, μ. That is equivalent to being unable to reject the null hypothesis at a significance level of 0.05. Conversely if the 100(1-a) confidence interval does not contain 1.15, then you reject the null hypothesis at the α level of significance.

5.3 Hypothesis Test Assumptions

The difference between hypothesis test procedures often arises from differences in the assumptions that the researcher is willing to make about the data sample. For example, the Z-test assumes that the data represents independent samples from the same normal distribution and that you know the standard deviation, σ. The t-test has the same assumptions except that you estimate the standard deviation using the data instead of specifying it as a known quantity.

Both tests have an associated signal-to-noise ratio

$$Z = \frac{\bar{x} - \mu}{\sigma} \text{ or } T = \frac{\bar{x} - \mu}{s},$$

where

$$\bar{x} = \sum_{i=1}^{n} x_i.$$

The signal is the difference between the average and the hypothesized mean. The noise is the standard deviation posited or estimated.

If the null hypothesis is true, then Z has a standard normal distribution, N(0,1). T has a Students t distribution with the degrees of freedom, ν, equal to one less than the number of data values.

Given the observed result for Z or T, and knowing the distribution of Z and T assuming the null hypothesis is true, it is possible to compute the probability (p-value) of observing this result. A very small p-value casts doubt on the truth of the null hypothesis. For example, suppose that the p-value was 0.001, meaning that the probability of observing the given Z or T was one in a thousand. That should make you skeptical enough about the null hypothesis that you reject it rather than believe that your result was just a lucky 999 to 1 shot.

There are also nonparametric tests that do not even require the assumption that the data come from a normal distribution. In addition, there are functions for testing whether the normal assumption is reasonable.

5.4 Example: Hypothesis Testing

This example uses the gasoline price data in $gas.mat$. There are two samples of 20 observed gas prices for the months of January and February, 1993.

```
>> load gas
>> prices = [price1 price2];
```

As a first step, you may want to test whether the samples from each month follow a normal distribution. As each sample is relatively small, you might choose to perform a Lilliefors test (rather than a Jarque-Bera test).

```
>> lillietest(price1)
ans =
     0
>> lillietest(price2)
ans =
     0
```

The result of the hypothesis test is a Boolean value that is 0 when you do not reject the null hypothesis, and 1 when you do reject that hypothesis. In each case, there is no need to reject the null hypothesis that the samples have a normal distribution.

Suppose it is historically true that the standard deviation of gas prices at gas stations around Massachusetts is four cents a gallon. The Z-test is a procedure for testing the null hypothesis that the average price of a gallon of gas in January (`price1`) is $1.15.

```
>> [h,pvalue,ci] = ztest(price1/100,1.15,0.04)
h =
     0
pvalue =
    0.8668
ci =
    1.1340    1.1690
```

The Boolean output is `h = 0`, so you do not reject the null hypothesis. The result suggests that $1.15 is reasonable. The 95% confidence interval [1.1340 1.1690] neatly brackets $1.15. What about February? Try a t-test with `price2`. Now you are not assuming that you know the standard deviation in price.

```
>> [h,pvalue,ci] = ttest(price2/100,1.15)
h =
     1
pvalue =
  4.9517e-004
ci =
    1.1675    1.2025
```

With the Boolean result `h = 1`, you can reject the null hypothesis at the default significance level, 0.05. It looks like $1.15 is not a reasonable estimate of the gasoline price in February. The low end of the 95% confidence interval is greater than 1.15. The function ttest2 allows you to compare the means of the two data samples.

```
>> [h,sig,ci] = ttest2(price1,price2)
h =
     1
sig =
    0.0083
ci =
   -5.7845   -0.9155
```

The confidence interval (`ci` above) indicates that gasoline prices were between one and six cents lower in January than February.

If the two samples were not normally distributed but had similar shape, it would have been more appropriate to use the nonparametric rank sum test in place of the t-test. You can still use the rank sum test with normally distributed data, but it is less powerful than the t-test.

```
>> [p,h,stats] = ranksum(price1, price2)
p =
    0.0095
h =
     1
stats =
       zval: -2.5928
    ranksum: 314
```

As might be expected, the rank sum test leads to the same conclusion but is less sensitive to the difference between samples (higher p-value).

The box plot below gives less conclusive results. On a notched box plot, two groups have overlapping notches if their medians are not significantly different. Here the notches just barely overlap, indicating that the difference in medians is of borderline significance (Figure 5.1. (The results for a box plot are not always the same as for a t-test, which is based on means rather than medians.) Refer to Chapter 3, Section 3.9 Statistical Plots for more information about box plots.

```
boxplot(prices,1)
set(gca,'XtickLabel',str2mat('January','February'))
xlabel('Month')
ylabel('Prices ($0.01)')
```

Available Hypothesis Tests

The Statistics Toolbox has functions for performing the following tests.

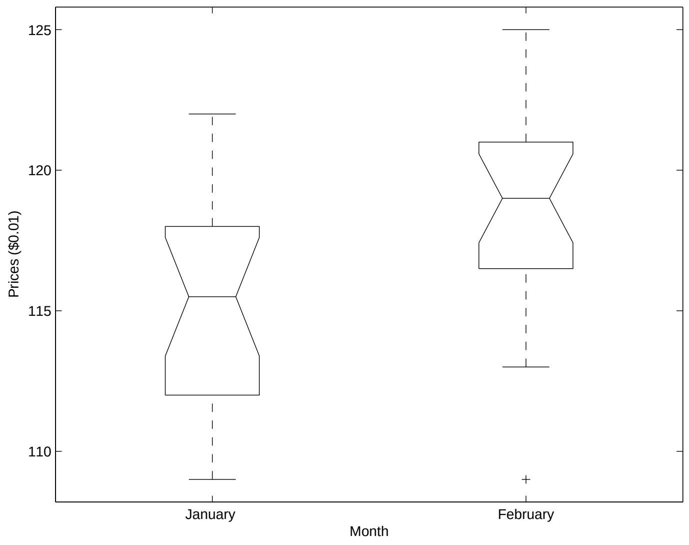


Figure 5.1: Boxplot

Function	What it Tests
`chi2gof`	Chi-square test of distribution of one normal sample
`jbtest`	Normal distribution for one sample
`kstest`	Any specified distribution for one sample
`kstest2`	Equal distributions for two samples
`lillietest`	Normal distribution for one sample
`ranksum`	Median of two unpaired samples
`runstest`	Randomness of the sequence of observations
`signrank`	Median of two paired samples
`signtest`	Median of two paired samples
`ttest`	Mean of one normal sample
`ttest2`	Mean of two normal samples
`vartest`	Variance of one normal sample
`vartest2`	Variance of two normal samples
`vartestn`	Variance of N normal samples
`ztest`	Mean of normal sample with known standard deviation

Appendix A

The First Appendix

The appendix fragment is used only once. Subsequent appendices can be created using the Chapter Section/Body Tag.

References [1] Bowman, A. W., and A. Azzalini, Applied Smoothing Techniques for Data Analysis, Oxford University Press, 1997.