

Data Analysis Lecture Notes Univ.-Prof. Dr. Peter Filzmoser Research Unit Computational Statistics Institute of Statistics and Mathematical Methods in Economics TU Wien Vienna, March 2024. The relationships between the different variables is usually most interesting and important. Small deviations from idealized conditions and assumptions could lead to false and misleading conclusions. An important goal in data analysis is to find procedures which are resistant to such deviations. The vast amount of data occurring today have heavily influenced the world of statistics. “Big data” is present in many application areas, and the use of traditional statistical methods is often prohibited. On the other hand, approaches such as Deep Learning seem to be more and more successful in real data applications. The concepts covered in this lecture will be demonstrated at data examples analyzed in the statistical software environment R. Most of the statistical procedures, even the newly developed ones, are implemented in R. The real challenge is to combine the different concepts for an improved analysis.

W.S. Cleveland (1993). Visualizing Data, Hobart Press, Summit, New Jersey. J.W. Tukey (1977). Exploratory Data Analysis, Addison-Wesley, Reading, Massachusetts.

1.1. Univariate scatterplots. 1 1.2. Histogram. 2 1.3 Density estimation. 3 1.4 Empirical distribution function. 4 1.5 Normal probability plot. 5 1.6. 1.6. Quantile-quantile plots%., and 1.7 Boxplots. 2. Statistical estimators for univariate data 19 2.1 “Classical” estimators of location and scale. 3. Bivariate density estimation. 4.1 Least-squares (LS) estimator. 4.2 Robust regression line after Tukey. 4,3, 4,4 Repeated median regression. 5 Estimation of non-linear trends. 6 Time series analysis. Modelling time series is one of the most common ways to predict future trends. This article includes a number of tools to help you with your time series modelling. Chernoff faces, stars, segments, Scatter diagrams. Profiles, stars,. segments, Chernoff faces%.,, -7.2, -8.1. The results are based on a multivariate analysis of data. The results are consistent with a robust estimation of covariance and correlation. The methods used to make the predictions are described in the next section. 10.1.

Partitioning methods. 10.2. Hierarchical clustering and fuzzy clustering. 10.3. Cluster validity measures. This chapter will NOT provide an overview about the various possibilities to visualize uni- variate data. Rather, we aim to discover WHY statistics is involved in visualizing univariate data, and HOW. Univariate scatterplots show the univariate data values on one axis. The points are scattered along the axis, thus the name scatterplot. This display already has difficulties in case of multiple points on one position, e.g. caused by rounding effects. Jittering: For every data value x_i we randomly generate a value y_i in a certain interval. Stacking: Multiple points are stacked on top of each other. The result is a barplot-like visualization.

1.2 Outliers Extreme outliers may dominate the scatterplot, and possible structure (groups, gaps, etc.) in the “data majority” might not be visible. Omitting those outliers can allow to see more details in the data majority. Dividing by the interval length is only important if the interval lengths differ. Usually, we work with equidistant histograms (equal interval lengths), and thus the denominator is irrelevant. The form of the histogram is heavily determined by the number of histogram bins. Rule of Sturges: The optimal number k for the optimal interval length $h_n = (t_k - t_1)/k$ is given by $k = \lceil \log_2(n) + 1 \rceil$. Rule of Scott: This rule is more general, and it assumes an underlying density function f which is continuous and bounded. Scott’s rule is sensitive to outliers. Rule of Freedman and Diaconis: In case of data outliers, the rule of Scott may lead to a non-optimal choice for the interval length. The idea is to use a more robust scale estimator, such as the interquartile range IQR. The density $f(x)$ at a location x is estimated based on observations which are in the interval $[x - h/2, x + h/2]$

Figure 1.2: Comparison of the rule of Scott and Friedman-Diaconis for normally distributed data without and with outlier.

	n	Scott	Diaconis
without outlier	10	1.620	1.252
	20	1.286	0.994
	30	1.123	0.868
	40	1.020	0.789
	50	0.947	0.743
with outlier	75	0.743	0.743

75. Density estimation is often called kernel density estimation (kde) because the resulting form depends on the choice of the weight function, also called kernel. Examples are: a) Rectangular weight function (boxcar function): $W(t) = \frac{1}{2} \left(1 + \frac{t}{|t|} \right)$

$W(x - x_i/h) = 1$ for $|x - x_i| \leq h/2$ and 0 otherwise. Figure 1.4 shows the resulting density estimates by using different weight functions. The resulting function $\hat{f}(x)$ is smoother, which might be preferable in practice. The choice of the window width h is obviously also crucial, and there exist different proposals in the literature.

Figure 1.4: Density estimation of the ozone data from Figure 1.1, with boxcar function (left) and cosine function (right). The optimal window width is proposed by the default of the R function. $\text{optimal window width} = 1.76 \cdot \text{IQR}(\text{ozone}) / n^{1/5} \approx 1.76 \cdot 10.0 / 250^{1/5} \approx 1.76 \cdot 10.0 / 3.98 \approx 4.42$.

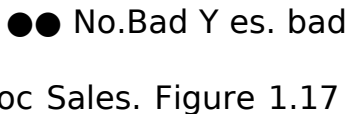
Figure 1.6 shows data from geochemistry. About 600 soil samples on the Peninsula Kola have been taken from different soil layers. The data sets are available in the R package `StatDA`. One can show that for $n \rightarrow \infty$, the empirical distribution function converges to the theoretical one. Figures 1.5 and 1.6 compare the empirical distribution function with the theoretical one. The idea of the normal probability plot is to rescale $F_n(x)$, such that the S-shape transforms to a line. Deviations from a line are easier to grasp. The underlying data distribution is F , which is here the distribution function of a normal distribution $N(\mu, \sigma^2)$. The idea is to plot x against $\Phi^{-1}(F_n(x))$, which is the definition of a linear function in x . At a probability of 50%, cutting the line formed by the points, we read on the other axis an estimate of μ .

Figure 1.7 shows the normal probability plots from the Kola data used in Figure 1.6. The rounding effect is again visible, but this time we can see every single data point. The left plot is for Sc , with quite big and systematic deviations from the line, indicating an underlying distribution different from the normal distributions. The quantile-quantile (or simply QQ-) plot compares an empirical distribution function, computed from the data sample, with a hypothetical one, e.g. with normal distribution. A fixed probability p defines the quantiles $q_x(p)$, $q_y(p)$ for both distribution functions. This is done for several probabilities, and the resulting quantiles are forming the data pairs on the horizontal and vertical axis. Figure 1.9 shows a QQ-plot for a data set with annual snowfall accumulations in Buffalo, NY, from 1910 to 1973. If the reference distribution F_x is the

standard normal distribution, then the quantiles $q_x(p_i)$ are computed in R. If F_y refers to the distribution of the data, the corresponding quantiles are simply the sorted data values. Figure 1.9: QQ-plot of the snowfall data from Buffalo (1910-1973), with the estimated parameters from the normal distribution. Relative frequency 20 40 60 80 100 120 140 0.000 0.005 0.010 0.015 Theoretical $N(80, 25^2)$ Density estimation

In Figure 1.6, the points should fall on a line. The intercept of the line is an estimation for μ , and the slope is a density estimation. We can expect bigger deviations from the line if the sample size is small. However, we are also interested in systematic deviations because this indicates that the data distribution is from a different family. Figure 1.10: Simulated data examples with a distribution that are different from a normal distribution. We can see deviations in the histograms, but particularly systematic deviation in the QQ-plots. In a QQ-plot we can expect a certain variability of the points, especially for small and big quantiles. One can construct point-wise confidence envelopes based on the scale estimation of the empirical quantile. This can be done by making use of the asymptotic behavior of order statistics. Figure 1.11 shows QQ-plots with additional confidence envelopes. The left plot is for the Kola C-horizon Sc data. They have even been square-root transformed first to better approximate normality. Still, there seems to be a systematic deviation, especially in the lower part of the distribution. Box plots are based on 'simple and robust ingredients'. This excludes the arithmetic mean and the sample variance. A common definition is graphically shown in Figure 1.12. Boxplots are very informative: Estimated center, by the median (robust!), Estimated scale, by IQR (Robust!). Visual impression of the data skewness, by position of the median inside the box. Judgement about possible data outliers, by relationship of length of box to length of whiskers. For normally distributed data, the boxplot outlier rule would (wrongly) "declare" 0.7% of data as outliers. Using order statistics one can construct a confidence interval (CI) around the median. The CI is shown in the boxplot by so-called notches. Notches for Boxplots: By making use of the theoretical properties of order statistics, can

see huge outliers. Figure 1.14 shows the corresponding boxplots, left plot for the traditional version, right plot with notches. In this context we would use the notch information as follows: If the notches of two boxes (for one measurement) do not overlap, then we would have strong evidence that the medians of the two species differ for this measurement. Figure 1.15 shows that the median sales unit in rural or urban regions is not significantly different (left), but there are pronounced differences. The right plot is with notches. Figure 1.16 reveals that the best (median) sales units are for shops located in the US, where the carseats are well visible in the shop. The plot below goes even further: in addition



16 ●● No.Bad Yes. bad No.Good Yes good No.Medium YesMedium 0 5 10 15 US:ShelveLoc Sales. Figure 1.17 shows that shops in urban areas have better sales units than those in rural areas. R package laeken contains the data set eusilc with synthetically generated data from real Austrian EU-SILC (European Union Statistics on Income and Living Conditions) data. There is information about the net income of employees, and several additional information on household size, sex, citizenship (Austria, EU, Other), and many more. The previous chapter mainly focused on statistical graphics in order to get an idea about the data generating process. In this chapter we mainly focus on estimating these two parameters by using a data sample. The arithmetic mean is the least-squares (LS) estimator in the location problem. Let X_1, \dots, X_n be independent and identically distributed (“i.i.d”) The arithmetic mean has also a disadvantage: It is not robust against outliers. In fact, if we would only move a single observation, the estimator will also move in the same direction – in the worst case as far as we want. The “classical” estimator of scale is the empirical standard deviation. The arithmetic mean was identified as the LS estimator in the location problem. The median is the so-called L1 estimator, minimizing $\sum_{i=1}^n |x - X_i|$. Under normality, the variance of \bar{x} is about 50% higher than that of \bar{x} . In other words, in order to attain the same precision, we would need about 1/3 more data for \bar{x} . 20 For a consistent estimator we would like to have (with probability 1) that the estimated

parameter comes very close to the true one, even if the sample size gets arbitrarily large. Robust estimators of scale are for example: Interquartile range. Scale estimator is defined as $MAD = \text{median } 1 \leq i \leq n |x_i - x_M|$, thus as the median of the absolute deviations from the median. Under normality, its variance is roughly 2.7 times higher than that of the empirical standard deviation. Q_n is a consistent estimator for σ under normality only with a correction factor: $sQ_n = 2.219 \cdot Q_n$ Example: Here a simple example how to compute the estimators. Sample: 2.1 3.7 2.6 5.8 1.6 1.1 32.7 4.7 3.1 4.8 Ordered sample: 1.3 1.4 1.8 2.3 2.4 29.3 $|x_i - x_M|$ (i): 0.3 0.2025, 0.0225, 1.32, 2.10, The term “outlier” has been mentioned already several times, but so far no “definition” has been provided. What is an outlier? From a statistical point of view we would have a data generating process in mind, which generates our “regular” data. But the outliers have been generated from a different process. When observing univariate data it would usually not be feasible to estimate the locations and scales of the two (or more) data generating distributions. Rather, we estimate the parameters for the distribution which generated the “data majority”, and then try to come up with an appropriate outlier cutoff. Outliers would be identified as observations which are smaller than $\hat{q}_{0.25} - 1.5 \cdot IQR$. Outliers can be identified by using the boxplot rule and the clean data rule. From theory we know that the probability to be outside of $\mu \pm 2 \cdot \sigma$ is 4.6%. However, the sample size is varied from 10 to 10.000 observations, which has an effect on the estimated parameters. We know from theory that the boxplot rule incorrectly identifies 0.7% of the observations as outliers. The “classical” rule seems to be quite useful for an outlier proportion of up to 15%, but this depends very much on how the outliers are simulated. The boxplot rule is robust against 25% of contamination, as it can be expected from theory. Scatterplots show the pairs of values directly in a coordinate system, with a horizontal and a vertical axis. Figure 3.1 left shows an example of a scatterplot, and the right plot also shows the marginal distributions of the variables with histograms and density functions. Multiple points in

scatterplots are possible in the bivariate case. There are various options to cope with this situation. Jittering (for one variable) can be done by the function `jitter()` in R. Table 3.1 shows a data set of managers employed at Bell Laboratories. reported is their age in the year 1982, and the number of years since they finished their studies. original data, and add as “background” a bivariate density estimation. Figure 3.2 shows the data which essentially reveal a linear relationship between both variables. The data set contains multiple points mainly because of rounding artifacts. Two clusters with higher point-density seem to indicate very active periods for hiring managers. In the 2-dimensional case it is also possible to estimate the density function. The basic principle is the same: select some local region – this time in two dimensions – such as a square or a circle. The weight function needs to be chosen such the the integral is 1. The density estimation is obtained at a grid defined by the intersection of a number of horizontal and vertical points. In R this can be done by the function `kde2d()` of the package MASS. The function returns the grid points along the two axes, and a matrix with dimensions given by the number of grid points. The R package MASS contains the data set `geyser`, with data from the “Old Faithful” geysers in the Yellowstone National Park. Two variables have been observed in the period August 1-15, 1985: the eruption time (in minutes) and the waiting time for this eruption (in minutes) The density estimation has been done with a bivariate normal weight function. Figure 3.3: Density estimation with the `geyser` data. Figure 3.4: Three-dimensional presentation of the density estimation.

Chapter 4: Estimation of linear trends. The goal is to estimate a linear function which allows to predict y based on given x information. This linear function $f(x)$ can be defined with two parameters, an intercept α and a slope β . With the estimated parameters we also obtain the fitted response. Least-squares (LS) estimator is the most widely used option which is also taught in school and in many courses at university. As the name suggests (but not very clearly!), the objective function to be minimized is the sum of squared residuals. This minimization problem has a unique

solution for the estimated parameters. The solution is even given by explicit formulas. The residual variance σ^2 is also an important parameter to be estimated. This is used to construct confidence intervals and statistical tests. LS estimator is unbiased, and among all unbiased estimators it has the smallest possible variance. If the mentioned assumptions are violated, the confidence intervals and statistical tests for the LS estimator can be misleading. If there are severe outliers, this estimator could be far from indicating the linear relationship between x and y . LS regression involves every single observation (with the same weight) in terms of the squared residuals. The square makes outliers even more dominating. From a robustness point of view we would like to fit a model to the data majority, but we would not necessarily like to accommodate every single observations. Simple method, developed by John Tukey (1970), estimated the parameters of the regression line. Sort the data pairs (x_i, y_i) according to their x -values. Compute medians of the x - and y -values in the single groups. Steps 3-5 with data pairs (x_i, y_i) for $i = 1, \dots, n$ yields line $\hat{y} = \hat{\alpha}_1 + \hat{\beta}_1(x - x_M)$ and residuals $r_i := y_i - \hat{\beta}_1(x_i - x_M)$. After the third iteration the line does not change any more. The breakdown point of the Tukey regression estimator is $1/6$. The breakdown point of the Theil estimator can be derived as follows. We need to identify the smallest number k of observations that is sufficient to cause breakdown of the estimator. The repeated median regression estimator is also based on medians of pair-wise slopes. This estimator was the first one to achieve a breakdown point of 0.5 . The outer median would lead to a non-sense result if at least $n/2$ values from the inner median are non-sense. This concept can be extended to the case where we have more input variables x_1 . The LMS regression estimator has been introduced by Rousseeuw (1984). The solution can be approximated by a so-called subsampling algorithm. The selected solution is the one with the smallest value of the LMS criterion. 6 Least Trimmed Squares (LTS) regression. The LTS estimator minimizes the sum of the smallest squared residuals, i.e. a trimmed sum. The parameter h determines the breakdown point, which

is in between 0% and 50%. R code: The package robustbase offers many robust procedures, not only for regression. LTS regression is implemented in the function `ltsReg()`. This also works in case of more than one input variable. Another function, `lmrob()` for MM regression, might even be preferable. In the more general case we want to predict an output variable y by using several inputs x_1, x_2, \dots, x_p . For example, if the response y refers to something like product quality (in a production process), then there could be several input variables which could be influential for good or bad quality. We assume again a linear relationship of the inputs with the response, and thus a model of the form: $y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p + \varepsilon$. The LS criterion would be natural, with the only drawback of the sensitivity to outliers. The leverage points would be outliers in the p -dimensional space of the input variables. We use simple linear regression with `Favor` as a predictor. The input variables are `flavor`, `appearance`, `taste`, `stickiness`, and `toughness`. We compare the results of LS regression with those of MM regression. The estimated regression parameters from both methods are very similar to each other. Figure 4.3 presents the resulting fits from classical and robust regression by using the two input variables `Favor` and `Appearance`. There is only a little difference in the results of the two methods. The problem can no longer be visualized. `Flavor`, `Taste` and `Stickiness` positively contribute to the `Overall` evaluation. The remaining columns are used for hypothesis tests: we test if the regression parameter is zero. The last column is the p -value, and if $p < 0.05$ we can reject the null hypothesis of a zero coefficient. For MM regression we obtain the following results:

```
re2 <- lmrob(Overall_evaluation ~ ., data = rice)
summary(re2)
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.09841	0.03729	-2.639	0.00966 **
Flavor	0.21569	0.07926	2.721	0.00769 **
Appearance	0.02917	0.71572		
Taste	0.60889	0.09639	6.317	7.64e-09 ***
Stickiness	0.36465	0		

Figure 4.4 shows diagnostics plots for MM regression which are obtained by `plot(re2)`. The left plot shows \hat{y}_i against y_i . The values are very similar to each other, which was already suggested by the high value of the multiple R-squared.

The outlier 75 deviates slightly. The following sections treat the problem of smoothing a signal. Trend estimation can be influenced by outliers, and thus a focus is on robust non-linear trend estimation. The main idea is to use an “algorithm” S for smoothing the time series values x_t . When applying S on x_t we obtain the smoothed values z_t . A linear filter can be very sensitive to outliers. An idea for such an algorithm is: $z_t = \frac{1}{4} x_{t-1} + \frac{1}{2} x_t + \frac{1}{4} x_{t+1}$. In many algorithms, the window of time points used for smoothing is centered at the time point where it is done. Smoothing by the median with an even number of $w = 2s$: $z_{t+1/2} = S(x_t) = \text{median}(x_{t-s+1}, \dots, x_t, x_{t+1})$. It is usually desirable that the smoothed signal has the same number of time points as the original signal, but median smoothing would lose time points on both ends. Figure 5.1 shows a time series with body temperatures of a cow. The temperature has been measured at 6:30am on 75 consecutive days. Non-linear robust smoothing will produce a smoothed signal which is hopefully also not much affected by outliers. In a second step one can inspect the residuals in order to identify potential outliers, this second step is called filtering. With non-linearity we can no longer assume constant residual variance. Rather, the residual variance needs to be estimated locally (and robustly). It is unclear how to do this with simple median smoothing. Nice solutions to this problem are implemented in the package `robfilter`. Let us assume an underlying but non-observable signal μ_t , for $t = 1, \dots, T$, which we want to estimate based on the given data. In fact, the observed data are generated according to the model $x_t = \mu_t + \epsilon_t$. For the outlier diagnostics we need an estimation of the median estimator. In the time window around t we obtain the residuals. residual scale σ_t at every time point t . This can be obtained with the above principle of local linearity. Based on this scale estimation we can use the standard procedure for outlier diagnostics: An outlier is identified if $|\hat{r}_t| > 2$. `Scarm` contains an algorithm which automatically adjusts the window width to the given data, based on statistical tests. This algorithm needs a lot more data. The key idea is to use locally weighted regression, where the regression model is either linear or

quadratic. regression coefficients are estimated by a weighted residual sum of squares, where the weights consider the local aspect (in x direction) and robustness aspects (in y direction) The algorithm is described below, here for the linear model. We fix a parameter s , called span, defining the proportion of data points to be used for local smoothing. The algorithm is implemented (with some more functionality) as function `loess`. Here is an application to the cow. Want to smooth at index i , thus in a local neighborhood of (x_i, y_i) , to obtain the smoothed value \hat{y}_i . Compute all distances to x_i along the x-coordinate. The `loess()` function can also be used for extrapolation. In that case, the function needs to be applied as follows:

```
res2 <- loess(Temperature~time, data=df, control = loess.control(surface = "direct"))
timeext <- predict(res2,data.frame(time=timeext),se=TRUE)
plot(df,xlim=c(0,100),ylim=c(36.5,40) lines(timeext,pred2$fit,col=3) timeext, Pred1$fit+2*Pred1$se.
```

Figure 5.4 shows the scatterplot matrix of the data, together with the smoothed lines from `loess()`. This plot can be produced with the function `scatterplotMatrix()` from the package `lattice`. The estimated standard errors are shown in addition as bands around the smoother lines. The goal of this chapter is to provide an overview of different approaches to analyzing data that depend on time. The form of the dependency is also crucial for the way how we analyze the data. Are there trends or seasonal changes? If so, how can we analyze them? Is there still remaining structure in the residuals, which are defined as signal minus trend and seasonality? Is it possible to do forecasting (predict into the future)? Are there structural breaks in the time series, or outliers? We will not cover all these topics, but provide insight into some. Figure 6.1. graphically. We can see a clear trend, but also clear seasonal patterns. A more detailed look at the data shows that the deviations towards higher values are somewhat bigger than those towards lower values. This could be a disadvantage later on once we want to do estimations. The previous example has shown trends and seasonality. In the following we will work with these log-transformed values. A trend means that the time series

increases or decreases for a longer period of time. Seasonality refers to certain regular variations. The method `stl` works according to the following scheme. For the beer time series we have $P = 12$ because we have monthly data. Since the length of the time series is T , we have $C = \lfloor T/P \rfloor$ cycles. Figure 6.2 tries to provide more insights into this procedure. The plot on top shows the detrended time series, thus the result after Step (1) The bottom plot collects all monthly time points in order to apply Step (2) This plot is also called cycle plot, available in R as `monthplot()` The results in Figure 6.3 do not reveal clear patterns for the remainder, but still some “peaks” of varying magnitude which could be of interest to the modeler. The next section shows some attempts to modeling time series which are not recommended. The coefficients can be estimated with the methods from Chapter 4.2. In the following we will use LTS regression. The linear model is of course too simple, and thus we should proceed with a more complex model. Figure 6.4: Linear model (left) and model with a squared term (right) for beer time series. The result is presented in Figure 6.4 (right) We obtain a smooth estimation of the trend. This could already be useful for a simple forecast of future values. Since the model is still relatively simple, the forecast will only be of limited use. LTS regression for new model `lines(t,res$fit)` Figure 6.5 shows the result, which is still far from being “perfect” for forecasts. We could still include terms of higher order to improve the fit, which means that we have to estimate even more parameters. All coefficients except β_4 are significantly different from 0.232. Since we have a representation with Fourier coefficients, both the sine and the cosine should be used in the model. Exponential smoothing provides smoothed values at every time point. The smoothing value at t depends on the actual value and on the smoothed value from time point $t - 1$. The smaller α is chosen, the smaller is the contribution of the most recent values. The weights into the past decrease exponentially, which justifies the name of the method. The forecast of all future values does not depend on the horizon h , for which we would like to have our prediction. This forecast is useful for stationary time

series, which neither have trend nor seasonality. It is not useful for time series with trend (or seasonality) For the beer time series we can use the method of Holt-Winters in R as follows. Here we consider both trend and seasonal component. The result of the smoothing method is shown in Figure 6.6 (top). The estimated parameters are: alpha: 0.07532444 beta : 0.07434971 # parameter for the trend gamma. #parameter for seasonality. Autocovariance of order k is defined as $Cov(x_t, x_{t-k})$, and it can be estimated by $ck = \frac{1}{T} \sum_{t=k+1}^T (x_t - \bar{x})(x_{t-k} - \bar{x})$. Stationary time series with autocorrelation zero for $k > 0$ are named as white noise. Figure 6.7 shows the water level (annual mean, in feet) of lake Huron in the period 1875- 1972. It is clear that there must be a time dependency, and this can also be seen in the plot. However, it is not at all clear how long this time dependency lasts. Correlations between neighboring time points can be transferred. The resulting correlation between x_t and x_{t-k} is thus influenced by the observations in between these time points. Figure 6.8 (right) shows the partial autocorrelation function (PACF) of the lake Huron data. The ACF and the PACF are important for selecting an appropriate model for time series. Below we list some of the basic models which are already useful for many applications. However, there exists a multitude of different models designed for specific application. A stationary time series follows an autoregressive process of order 1, denoted as AR(1) The partial autocorrelations of AR(p) are 0 for lags bigger than p. An example of a simulated AR(2) process is shown in Figure 6.9. If neither the ACF plot nor the PACF plot show an “implosion” to zero from a certain lag on, the process could be a mixture of AR(p) and MA(q), denoted as ARMA(p, q) ARMA models require a stationary time series. Time series with a trend can be eliminated by using differences. In R we can estimate the parameters by `arima(data,order=c(p,d,q))` for the time series data, with the corresponding orders of the ARIMA(p, d, q) model. Parameter estimation for the above models is done according to the least-squares principle, thus by minimizing $\sum_{t=1}^T (x_t - \hat{x}_t)^2$. We have chosen an ARMA(1,1) model, and R gives back the estimated

parameters. We have no idea whether the model is appropriate. A crucial step for model selection is diagnostics. Diagnostic plots can be generated in R by: `tsdiag(fit)`

Figure 6.10: Diagnostic plots for the ARMA(1,1) model applied to the lake Huron time series. middle plot is the ACF plot of these residuals, and it should not contain any significant correlations. bottom plot presents the Ljung-Box statistic with p-values up to lag 10. Overall, our model seems to be well specified. Figure 6.11 shows the forecast for the next 8 years, jointly with a 95% confidence interval. The width of this interval reveals that – although the model seems to make sense – it is not really valuable for prediction. Figure 6.12 shows historical and more recent data of the annual mean water level of lake Huron. The forecast is – if at all – only useful for a very short period of time. Our model either seems to be too simple, or one can simply not find a better model. The confidence interval covers essentially the whole data range. The forecast quickly stabilizes to the mean.

Figure 7.1: Iris data: the four variables are represented in the plot by encoding two of them in symbol size and color. Another possibility is to present all $p \geq 2$ variables in a scatterplot matrix, see Figure 7.2. One can quite well see that the points fall into 2 clusters, but since the observations originate from 3 species, they should rather form 3 clusters.

Figure 7.2: Iris data: scatterplot matrix, with colors for the different Iris species. In principle one could use any projection directions, and those plots could be better suited to reveal group structures or patterns.

Figure 7.3 shows profiles of the different observations from Table 7.1. We could also transpose this table, and then the observations (profiles) would be the population groups. The example we have presented is a very specific type of a profile plot. We construct for every observation a star-type symbol, by arranging the p variable axes radially in equal angles. The (scaled) values of the observations are drawn as distances from the origin, and those resulting points are connected by lines.

Figure 7.4 shows star plots for the data set `data(mtcars)`, containing information of 32 different cars for 7 car characteristics.

Figure 7.4: Representation of the car data set by star symbols. (stars)

7.3 Segments Segments are very similar to stars. probably visually identify groups of similar cars. Figure 7.5: Representation of the car data by segments. (stars) 7.4 Chernoff faces The idea is to construct “faces”, and the different variables are encoded into the different characteristics of a face. Figure 7.6: Representation of the cars data by Chernoff faces. (faces from li- brary(aplpack) 69 7.2.5 Boxes Here the p variables are first grouped into 3 classes. The size of the box is determined by the relative proportion of the observations in the groups. Figure 7.7: Representation of the cars data set by boxes. (boxes aus library(StatDA) 7.3 Trees Trees try to express the correlation structure between the variables. The variables or variable groups form branches according to their similarity. Figures 7.8 and 7.9 show the car data set represented by trees. Castles can be viewed as very specific trees, where the “branches” are drawn with an angle of zero to the vertical direction. Figure 7.9: Representation of the car data by castles. Pantera LFerrari DinoMaserati BoraVolvo 142E qsec hp cyl disp mpg drat wt Figure 7.10 shows another car data set, available as data frameAuto. Covariance and correlation are possibilities to characterize the relationship between the variables. Theoretically, the covariance between a pair of random variables x_j and x_k is defined as $\sigma_{jk} = E$. The population correlation coefficient between x_j and x_k is defined as $\rho_{jk} = \sigma_{jk} / \sqrt{\sigma_{jj} \sigma_{kk}}$. It is a dimension-free measure for the linear relationship between these two random variables, which is always in the interval $[-1, 1]$. The correlation is easier to interpret than the covariance because it does not depend on the variances of the variables. For $j = k$ we obtain the sample variance, see Section 2.1. The classical estimator for the correlation coefficient ρ_{jk} is the sample correlation coefficient. If $r_{jk} = 0$, then there exists no linear relationship between x_j and x_k ; however, there could still exist a nonlinear relationship. In each of the plots we can see 200 observations, and they are generated from a bivariate normal distribution with correlation 0.8. The sample correlation coefficients will very likely be close to these theoretical values. More robust estimators of correlation would be

median and MAD. A “robustified” pairwise estimation of the elements of the correlation matrix might be problematic. Depending on the pair, one would probably have to downweight different observations. This could lead to an “inappropriate” robustly estimated matrix. One approach for a more robust estimation is the Spearman rank correlation. The MCD estimator is defined by that subsample of size h which has the smallest determinant of its sample covariance matrix. The robust correlation matrix is given according to the definition of a correlation, namely by c_{jk} . The Euclidean and Minkowski distances are based on the same data. Let us again consider observations in the p -dimensional space. We will focus now on the relationships between the observations. An important concept is the cosine of the angle α between the observation vectors. This measure is independent of the lengths of the vectors and thus only considers the relative values of the variables. Another important distance measure is the Mahalanobis distance. This distance measure accounts for the covariance structure of the data. Figure 8.3 compares the Euclidean distance (left) with the Mahalanobis distance (right). Every point on a circle (ellipse) has the same distance to the center, and the distances increase for circles (ellipses) further away from the center. If the variables are correlated, e.g. with the covariance matrix I , both distance measures lead to the same answer. The Mahalanobis distance is very useful for the purpose of identifying outliers in the multi-variate data space. Outliers could be observations which are “far away” along one coordinate, but those observations could also be close to the center of the distribution. Outliers can be identified even by univariate outlier detection methods. Outliers are observations which are not extreme along a coordinate, but unusual concerning the joint data distribution. In order to apply the Mahalanobis distance, we first have to estimate the location and the covariance matrix. Figure 8.4 presents an example for multivariate (bivariate) outlier detection. The data set `glass` from `library(chemometrics)` is used, consisting of measurements of 13 chemical elements (compounds) for 4. For simplicity we just consider MgO and Cl,

and only 2 glass types. 120 observations are from the first type, and 10 from the second. The 10 observations could have a different data structure and therefore represent multivariate outliers. Figure 8.4: Two-dimensional glass data for two glass types, and 97.5% tolerance ellipse based on classical (left) and robust (right) estimates of location and covariance. The corresponding robust distances are more reliable for the identification of the second glass type.

1 2 3 4 0.2 0.4 0.6 0.8 1.0 1.2 MgO Cl

Since the original data set has 13 variables, we now compute classical and robust Mahalanobis distances in this 13-dimensional space. Note that the outlier cutoff value is now $q \chi^2_{13;0.975} = 4.97$. The right plot with the robust distances reveals data subgroups in the majority class. The goal here is to down-project the data to low dimension, typically to two, in order to be able to “look” at the data. Dimension reduction will be achieved by using linear combinations of the original variables. U is a linear combination of the variables of the form $u = x_1b_1 + x_2b_2 + \dots + x_pb_p$. It is common to call the coefficients loadings, and the values u scores. The k combinations can also be interpreted geometrically. We can view the variables x_1, \dots, x_p as the axes of a p -dimensional space. A linear combination u would point at a particular direction in this space, defined by the contributions of the loadings.

Principal components

Principal Component Analysis (PCA) is considered as one of the most important methods in multivariate statistics. PCA is the “mother” of the multivariate methods, described in the following section. The main goal of PCA is dimension reduction. The PCs are defined via linear combinations (what a surprise) This time, however, we will use very specific loadings B , the PCA loadings, which result in very specific scores U . The issue with centering and scaling will be discussed in more detail. “Relevant” refers to “explained variance”, and this is the key to defining the PCA loadings and scores. Since the goal is dimension reduction, we would like that the most relevant information is already contained in the first few columns of U . Figure 9.1 shows for a very simple data set of 10 observations in two dimensions the first PC.

The direction is determined by the loadings vector b_1 , and the scores u_1 are the data points orthogonally projected on this direction. These new univariate data points have the largest possible variance among all possible projections. Maximizing variance under constraints can be easily formulated by a Lagrange problem. Since the PCs are ordered according to variance, we could use the first $k < p$ PCs to express the most essential information in fewer coordinates. The j -th PC is defined as $u_j = Xb_j$, for $1 \leq j \leq p$, and we want to maximize $\text{Var}(u_j) = \text{Var}(Xb_j) = b_j^T \text{Cov}(X) b_j$. The data matrix is involved in terms of $\text{Cov}(X)$, which refers to the covariance matrix, where we denote with "Cov" a specific covariance estimator. $\text{Cov}(X)b_j = \lambda_j b_j$ for $j = 1, \dots, p$, which is just the form of an eigenvalue-eigen vector problem. Eigenvalues are equal to the maximized variances of the PCs. This is now the final answer to the question how we shall select the matrix B . The goal of PCA is to reduce dimensionality, and at the same time to keep the loss of information small. Since the variances of the PCs are sorted in decreasing order, we can easily compute the variance expressed by the first k PCs. The PC number versus this proportion is visualized by the PC number. The optimal number k of PCs could then be selected at that point before the curve "flattens out", which is indicated here by the straight line. The desired percentage really depends on the subsequent purpose of the analysis. For exploratory data analysis purposes we might also be satisfied with a (much) lower percentage. Centering will only be relevant when we would use a different procedure to compute PCs, such as SVD. Scaling, however, always makes a difference. After scaling, the variances of the data columns are equal to one. Figure 9.2: Scree plot and cumulative proportion of explained variance. The data set is obtained from `data(scor, package="bootstrap")`. We typically do is to eigen-decompose the correlation matrix rather than the covariance matrix. This explains the argument `cor=TRUE` inside the `princomp()` function. Figure 9.3 shows a bivariate data set with body and brain weight of different animals (sorry, also the human is included). The data set is available as `data(Animals2)` from the R package

robustbase. With this principal component, 52.6% of the total variance are explained. We can conclude that a transformation of the data to approach normality had clear advantages in terms of sensitivity. Figure 9.3: Body and brain weight of various animals: left only scaled, right log-transformed and scaled. The first PC and the resulting explained variance are indicated in the plots. The left plot shows the first PC with the robustly scaled data (89.2% explained variance) and the right plot contains the firstPC for the robustly scaled log- transformed data (97.8% explained) In Figure 9.2 we have used the data set from the package bootstrap with information from 88 students about their results in the subjects ME (me- chanics), AG (analytical geometry), LA (linear algebra), AN (analysis), and ES (elementary statistics) In each subject they could attain up to 100 points. From the scree plot and the plot of the cumulative variances we know that $k = 2$ components might be sufficient for data. PCA for scaled data

```

data 87 0 10000 20000 30000 0 100 200 300 Body weight (scaled) Brain weight (
scaled) u1 PC1: 89.2% explained -2 -1 0 1 2 3 -2 -1 0 0 0 1 0 1 1 1 2 1 2 0 0 2 1 0 2
0 2 2 1 1 0 3 0 0.1 0.2 0.3 0.0 0.01 0.02 0.00 0.000 0.0000 0.003 0.001 0.006 0.008
0.007 0.018 0.019 0.005 0

```

The scores plot (left) reveals that the student IDs from 1 to 88 seem to be ordered along PC1. PC2 is not so easy to interpret, but we can see, for example, that students 66 and 76 with intermediate average results must be quite good in ME and AG, but poor in AN and ES. “approximation” because we lost some information; in this example about 20%, because we reduced dimensionality from 5 to 2.

```

biplot(pca) # biplot with the princomp object 88 -4 -2 0 2 4 -2 -1 0 1 2 Comp.1
Comp.2 ME AG LA ANES Loadings

```

Figure 9.5: Scores (left) and loadings (right) of the first two PCs for the data set `data(scor,package="bootstrap")`. The term “cluster” has the meaning of a “concentrated” group. The task is to automatically group the observations into “homogeneous” groups. The observations within a group should be similar to each other. cluster analysis involves looking for groups of similar variables. The similarity of observations can be determined by a distance measure. Most of the

distance measures depend on the scale of the variables, and thus centering and scaling the variables is often an important first step. There are different procedures to construct clusters.

Partitioning methods: The observations are grouped into k clusters, and thus every observation is assigned to exactly one cluster.

Hierarchical clustering methods: A hierarchy of partitions is constructed, where the number of clusters is varied.

10.1.1 Partitioning methods

Suppose we want to partition the observations into k distinct clusters. For partitions we have that $n_1 + n_2 + \dots + n_k = n$. The most popular algorithm is the k -means algorithm. k -means is $k \times \sum_{j=1}^k n_j \times \sum_{i \in I_j} \|x_i - \bar{x}_j\|^2 \rightarrow \min$, (10.2) and it has to be minimized in order to find the resulting index sets with the cluster assignments and the corresponding cluster centers. The algorithm follows an iterative scheme, where in every step of the iterations the index sets I_j will (slightly) change. After convergence we obtain the final index sets, and therefore the final assignments of the n observations to the k clusters. An algorithm which is very similar to k -means is called PAM (Partitioning Around Medoids). Here the cluster centroids are robustly estimated by medians. The decision on an appropriate number k of clusters is usually done with a validity measure. In this article we will describe the procedure for agglomerative clustering in hierarchies. At the beginning, every observation forms an own cluster; these n clusters are called singletons. In the next step, those singletons with smallest distance to each other are merged, and we obtain $n - 1$ clusters. Hierarchical clustering algorithm is known for the 'chaining effect', which means that clusters typically grow by adding small clusters in every step. In contrast to k -means, this algorithm does not result in any randomness. The results in Figure 10.2 can be visualized more effectively in the so-called dendrogram. Every observation forms its own cluster. These are then merged step-by-step, until all observations are joined in one single cluster. Horizontal lines refer to connections of clusters at the corresponding distance. The observations are arranged in order to avoid intersections between the lines. Fuzzy clustering allows for a "soft" assignment – it is often not so clear if an

observation belongs to one cluster or falls “in between” several clusters. More specifically, partitions assign each of the n observations to precisely one of the k clusters. The most widely used algorithm is the fuzzy c-means algorithm. The number of clusters k is – similar to k-Means – given by the user. This is done by a membership coefficient u_{ij} , for $i = 1, \dots, n$ and $j = 1, \dots, k$. Fuzzy clustering is based on a p -dimensional normal distribution. The underlying data distribution is considered as a mixture of k such normal distributions, with prior probabilities p_j , summing up to 1. For different runs. Once the cluster assignments are estimated, one can re-estimate the parameters of the normal distributions as well as the prior probabilities. This procedure, iterating the two steps, is called EM (expectation maximization) algorithm. The main idea is to reduce the number of parameters of the covariance matrices to be estimated. The very simplest option would be $\Sigma_j = \sigma^2 I$, for $j = 1, \dots, k$. Figure 10.4 illustrates different restricted models. In the example we use the function `Mclust()` from the package `mclust` to cluster the iris data set. This algorithm allows to provide a range of different numbers of clusters, here from 3 to 9. Results are shown in Figure 10.5. The aim of cluster analysis is to end up with homogeneous clusters, i.e. the observations assigned to the same cluster should be very similar to each other. A cluster validity measure should help with the decision about an appropriate number of clusters k . EII VII EEI VEI EVI VVI EEE EEV V EV VEV VVV Sepal.Length Petal.Width Sepal.-Petal.length Sepal-Petal.-Length Sepal - Petal.-Sepal. width Petal-Sepal-length Sepa-Sepa-Length Sepa - Sepa. length Sepa: Petal.: Sepa; Sepal.: Petal.; Sepa : Petal., Petal: Sepa, Petal.'s length, Sepa's width, Peta'sLength: Peta.Length; Peta: Measurements of homogeneity could be based on the number of clusters. The scatterplot matrix with assignments and ellipses for the best solution should be dissimilar to each other. cluster distance measures complete linkage, single link- age, etc., or we can consider the between-cluster sum-of-squares B_k . The values of B_k should preferably be big, but again this depends on the number of clusters. The Calinski-Harabasz index is a normalized

ratio of these quantities. For Calinski-Harabasz we would take that k which gives the biggest value. 97 For Hartigan we would taken that k where we can see a “knee” in the plot, thus where the increase flattens out. We admit that these instructions might not be entirely clear in a specific application. The goal of discriminant analysis is also very different from cluster analysis. Here we want to use the available information to establish discriminant rules, which later on will enable that new test set observations can be “discriminated” We assume that the groups have prior probabilities p_j , where $p_1 + \dots + p_k = 1$. We would then assign the observation to that group for which we obtain the biggest value of the posterior probability. If we just compare the posteriors of two groups, we could look at the (logarithm of the) ratio. The discriminant rule is indeed a linear function in x , which justifies its name. x is assigned to the j -th group (and not to the l -th groups) if $\delta_j(x) > \delta_l(x)$, where δ_j is called linear discriminant function of the j th group. In order to apply the rule (10.10), we first need to estimate the unknown parameters from the training data. The prior probabilities p_j can be estimated by the group proportions n_j/n . If we do not want to assume equal group covariances, then we will see that plugging in (8.5) into (9) will give the same result. In QDA we need to estimate many more parameters than in LDA. This could lead to an overfit of the discriminant rule to the training data, with the risk of a poorer performance for the test data. In contrast to cluster analysis we now make use of the grouping information to estimate the discriminant functions. If we would evaluate the performance on the same “training data” (and we just have a single data set), we would get a “too optimistic” impression of the classifier. This can be avoided by first randomly splitting the data into training and test set (which could be done several times) Fig. 10.6 shows a projection of the data into the space of the linear discriminant functions. Textual symbols are referring to the training data, and color corresponds to the groups. The test data set is represented by non-textual symbols; color correspond to the true group membership, symbol to the predicted one. Statistics is different from detective work, where we would

search for 'any' hints. Data need to be comparable if they should be analyzed jointly. In the course of time the general framework could change. A sample is a subset of a population within a statistical survey. The sample should be composed in a way to investigate certain properties of the whole population. Based on the sample we want to draw conclusions on the population. The first principle is to obtain a sample which is representative. A representative sample needs to reflect the complexity and the composition of the population. If we are interested in the mean household income, the sample must consider the different characteristics of the structure of a population. The selection according to Figure 11.2 can only be based on a quota sample. The population consists of all computers (IP addresses) which accessed their products. The statistical units to be measured could consist of all accesses from IP addresses to. We distinguish among three different types of data: Primary data collection: We ourselves are collecting the data. The type of the survey is internet-based (and not based on questionnaires) The amount is defined by the selected time interval and products. Secondary data collection: We make use of existing data bases, such as data bases from Statistics Austria or EuroStat. Data bases from banks, insurance companies, other companies, etc. accessible, and that manipulation by the interviewer ("leading questions") has to be strictly avoided. Missing values are a problem for many statistical methods. Data could contain outliers, which can either be corrected, or if this is not meaningful, we need to reduce their influence. For some statistical methods it might be necessary to do data transformations first. A deeper mathematical understanding of the methods is always useful and recommended. More general content-wise interpretations should be done by the domain experts, ideally with support from the data analysts.